

MACHINE LEARNING FOR ALGORITHMIC TRADING

4 BOOKS IN 1

Master as a PRO applied artificial intelligence and Python for predict systematic strategies for options and stocks. Learn data-driven finance using keras

**JASON TEST
MARK BROKER**

VISIT...

LANZAROTE
Caliente.COM

APPRENDIMENTO DELLA MACCHINA PER ALGORITMICO COMMERCIO

4 LIBRI IN 1

Impara l'arte della programmazione con un corso intensivo completo per principianti.
Strategie per padroneggiare Data Science, Numpy, Keras, Pandas e Arduino come un
Pro in 7 giorni

JASON TEST E MARK BROKER

© Copyright 2020 - Tutti i diritti riservati.

Il contenuto di questo libro non può essere riprodotto, duplicato o trasmesso senza il permesso scritto diretto dell'autore o dell'editore.

In nessun caso sarà ritenuta responsabile alcuna colpa o responsabilità legale nei confronti dell'editore, o dell'autore, per eventuali danni, riparazioni o perdite monetarie dovute alle informazioni contenute in questo libro. O direttamente o indirettamente.

Avviso legale:

Questo libro è protetto da copyright. Questo libro è solo per uso personale. Non è possibile modificare, distribuire, vendere, utilizzare, citare o parafrasare alcuna parte o il contenuto di questo libro senza il consenso dell'autore o dell'editore.

Avviso di esclusione di responsabilità:

Si prega di notare che le informazioni contenute in questo documento sono solo a scopo educativo e di intrattenimento. È stato compiuto ogni sforzo per presentare informazioni accurate, aggiornate, affidabili e complete. Nessuna garanzia di alcun tipo è dichiarata o implicita. I lettori riconoscono che l'autore non si impegna a fornire consulenza legale, finanziaria, medica o professionale. Il contenuto di questo libro è stato derivato da varie fonti. Si prega di consultare un professionista autorizzato prima di tentare qualsiasi tecnica descritta in questo libro.

Leggendo questo documento, il lettore accetta che in nessun caso l'autore è responsabile per eventuali perdite, dirette o indirette, sostenute a seguito dell'uso delle informazioni contenute in questo documento, inclusi, ma non limitati a, — errori, omissioni o imprecisioni.

SOMMARIO

MACHINE LEARNING CON ALGORITMO TRADING PYTHON PER PRINCIPIANTI

introduzione

- 1. Ottimizzazione del codice Python con ctypes**
 - 2. Trovare il toolkit perfetto: analizzare i modelli di progetti Python più diffusi**
 - 3. Come vengono implementati i tipi interi ampi in Python?**
 - 4. Creare un bot in Python per imparare l'inglese La**
 - 5. termocamera sul Raspberry PI Trovare un**
 - 6. parcheggio gratuito con Python Creare giochi sul**
 - 7. framework Pygame | Parte 1**
- Creazione di giochi sul framework Pygame | Parte 2**
- Creazione di giochi sul framework Pygame | Parte 3**

8. Programmazione orientata agli oggetti (OOP) in Python 3

Conclusione

PYTHON PER LA SCIENZA DEI DATI

Introduzione

Data Science e il suo significato

Fondamenti di Python

Funzioni

Liste e loop

Aggiunta di dati con più valori in Python

Aggiunta di dati stringa in Python

Dati del modulo

Conclusione

TRADING DI OPZIONI PER PRINCIPIANTI

Introduzione

- 1. Nozioni di base sul trading di opzioni**
- 2. Perché fare trading con le opzioni?**
- 3. Principali concetti di trading di opzioni**
- 4. Miti e malintesi sul trading di opzioni**
- 5. Strategie di trading con le migliori opzioni**
- 6. Le migliori qualità di un trader di opzioni di successo**
- 7. Come selezionare l'opzione giusta per un guadagno maggiore**
- 8. Suggerimenti e trucchi importanti sul trading di opzioni**
- 9. Domande frequenti importanti sulla conclusione del**

trading di opzioni

INTRODUZIONE AL

TRADING DAY E SWING

- 1. Differenza tra swing trading e day trading**
- 2. Come funziona?**
- 3. Swing trading e day trading: tattiche e strategie**
- 4. Indicatori di swing e day trading**
- 5. Pro e contro dello swing e del day trading**

conclusione

PITONE PER PRINCIPIANTI

Una guida al corso intensivo per l'apprendimento automatico e la programmazione web. Impara un linguaggio informatico in semplici passaggi con esercizi di codifica.

JASON TEST

INTRODUZIONE

I modelli di progettazione sono modelli riutilizzabili per risolvere problemi noti e comuni nell'architettura software.

T Sono meglio descritti come modelli per lavorare con una situazione normale specifica. Un architetto può disporre di un modello per la progettazione di determinati tipi di telai di porte, che inserisce in molti dei suoi progetti, e un ingegnere del software o un architetto di software deve conoscere i modelli per risolvere le attività di programmazione comuni.

Un'eccellente presentazione del modello di progettazione dovrebbe includere:

- Nome
- Problema motivante
- Decisione
- Effetti

Problemi equivalenti

Se pensavi che fosse un piuttosto [vago](#) concetto, avresti ragione. Ad esempio, potremmo dire che il seguente "schema" risolve tutti i tuoi problemi:

- Raccogliere e preparare i dati necessari e altre risorse
- Effettuare i calcoli necessari e svolgere il lavoro necessario
- Registrare ciò che si fa
- Libera tutte le risorse
- ???
- Profitto

Questo è un esempio di pensiero troppo astratto. Non puoi chiamarlo template perché non è un ottimo modello per risolvere nessun problema, anche se tecnicamente è applicabile a nessuno di essi (compreso il cucinare la cena).

D'altra parte, potresti avere soluzioni troppo specifiche per essere

chiamato modello. Ad esempio, potresti chiederti se [QuickSort è un modello](#) per risolvere il problema di ordinamento.

Questo è, ovviamente, un problema comune del programma e QuickSort è una buona soluzione per questo. Tuttavia, può essere applicato a qualsiasi problema di smistamento praticamente senza modifiche.

Una volta che hai questo nella libreria e puoi chiamarlo, il tuo unico vero lavoro è rendere in qualche modo comparabile il tuo oggetto, e non devi occuparti della sua entità da solo per cambiarlo per adattarlo al tuo problema specifico.

Problemi equivalenti si trovano da qualche parte tra questi concetti. Questi sono problemi diversi che sono abbastanza simili da poter applicare loro lo stesso modello, ma sono abbastanza diversi che questo modello è significativamente personalizzato per essere applicabile in ogni caso.

I modelli che possono essere applicati a questo tipo di problemi sono quelli che possiamo chiamare significativamente modelli di progettazione.

Perché utilizzare modelli di progettazione?

Probabilmente hai familiarità con alcuni modelli di progettazione attraverso la pratica di scrittura del codice. Molti bravi programmatori finiscono per gravitare verso di loro, senza nemmeno essere esplicitamente addestrati, o semplicemente li prendono dai loro anziani lungo la strada.

La motivazione a creare, apprendere e utilizzare modelli di progettazione ha molti significati. Questo è un modo per denominare concetti astratti complessi per fornire discussione e apprendimento.

Rendono più veloce la comunicazione all'interno del team perché qualcuno può semplicemente usare il nome del modello invece di chiamare il consiglio. Ti permettono di imparare dalle esperienze di persone che sono state prima di te, e non di reinventare la ruota, passando attraverso l'intero crogiolo di migliorare gradualmente le pratiche da solo (e costantemente rannicchiato dal tuo vecchio codice).

Le decisioni sbagliate che di solito vengono prese perché sembrano logiche a prima vista vengono spesso chiamate [anti-schemi](#). Perché qualcosa possa essere giustamente chiamato anti-schema, deve essere reinventato e, per lo stesso problema, deve esserci un modello che lo risolva meglio.

Nonostante l'apparente utilità di questa pratica, la progettazione di modelli è utile anche per l'apprendimento. Ti introducono a molti problemi che potresti non aver considerato e ti permettono di pensare a scenari con cui potresti

non aver avuto esperienza pratica.

Sono obbligatori per la formazione per tutti e sono un'ottima risorsa di apprendimento per tutti gli aspiranti architetti e in via di sviluppo che possono essere all'inizio della loro carriera e che non hanno esperienza diretta nell'affrontare i vari problemi che il settore offre.

Modelli di design in pitone

Tradizionalmente, i modelli di design sono stati suddivisi in tre categorie principali: **creativo**, **strutturale**, e **comportamentale**. Esistono altre categorie, ad esempio modelli architettureali o modelli di concorrenza, ma esulano dall'ambito di questo articolo.

Esistono anche modelli di progettazione specifici di Python che vengono creati specificamente attorno ai problemi forniti dalla struttura del linguaggio stessa o che risolvono i problemi in modi unici che vengono risolti solo grazie alla struttura del linguaggio.

I modelli di generazione riguardano la creazione di classi o oggetti. Servono ad astrarre le specificità delle classi, in modo che siamo meno dipendenti dalla loro esatta implementazione, o che non abbiamo a che fare con costruzioni complesse ogni volta che ne abbiamo bisogno, o che forniamo alcune proprietà speciali dell'istanziamento. Sono molto utili per ridurre la dipendenza e controllare come l'utente interagisce con le nostre classi.

I modelli strutturali riguardano l'assemblaggio di oggetti e classi in strutture più grandi mantenendo queste strutture flessibili ed efficienti. Di norma, sono davvero utili per migliorare la leggibilità e la manutenibilità del codice, garantendo la corretta separazione delle funzionalità, l'incapsulamento e la presenza di interfacce minime efficaci tra cose interdipendenti.

I modelli comportamentali riguardano gli algoritmi in generale e la distribuzione della responsabilità tra gli oggetti interagenti. Ad esempio, sono una buona pratica quando potresti essere tentato di implementare una soluzione ingenua, come un'attesa frenetica, o caricare le tue classi con codice non necessario per uno scopo specifico, che non è il nucleo della loro funzionalità.

Modelli generativi

- [Fabbrica](#)
- [Fabbrica astratta](#)
- [Costruttore](#)

- [Prototipo](#)
- [Singleton](#)
- [Pool di oggetti](#)

Modelli strutturali

- Adattatore
- Ponte
- Composito
- Decoratore
- Facciata
- Peso mosca
- proxy

Modelli comportamentali

- Catena di responsabilità
- Comando
- Iteratore
- Mediatore
- ricordo
- Osservatore
- Stato
- Strategia
- Visitatore

Modelli di progettazione specifici di Python

- Modello oggetto globale
- Modello di metodo prebound
- Modello di oggetto Sentinel

Digita Annotazione in Python

Innanzitutto, rispondiamo alla domanda: perché abbiamo bisogno di annotazioni in Python? In breve, la risposta sarà questa: aumentare il contenuto informativo del codice sorgente e poterlo analizzare utilizzando strumenti specializzati. Uno dei più diffusi, in questo senso, è il controllo dei tipi variabili. Anche se Python è un linguaggio con tipizzazione dinamica, a volte è necessario il controllo del tipo.

Secondo PEP 3107, potrebbero esserci i seguenti casi d'uso di annotazioni:

- controllo del tipo:
- espansione della funzionalità IDE in termini di fornitura di informazioni sui tipi di argomenti previsti e sul tipo di valore restituito per le funzioni:
- sovraccarico di funzioni e lavoro con generici:
- interazione con altri linguaggi:
- uso nelle funzioni logiche predicative:
- mappatura delle richieste nei database:
- parametri di marshalling in RPC (chiamata di procedura remota)

Utilizzo delle annotazioni nelle funzioni

Nelle funzioni, possiamo annotare gli argomenti e il valore restituito. Potrebbe assomigliare a questo:

```
def ripetitore (s: str, n: int) -> str:  
    return s * n
```

Un'annotazione per un argomento è definita dopo i due punti dopo il suo nome:

```
nome_argomento: annotazione
```

Un'annotazione che determina il tipo del valore restituito dalla funzione è indicata dopo il suo nome utilizzando i caratteri ->

```
def nome_funzione () -> tipo
```

Le annotazioni non sono supportate per le funzioni lambda

Accesso alle annotazioni delle funzioni

Utilizzato nella funzione può essere ottenuto tramite l'attributo `__annotations__`, in

quali annotazioni sono presentate sotto forma di dizionario, dove le chiavi sono attributi e i valori sono annotazioni. Il valore principale restituito dalla funzione viene memorizzato nel record con il tasto Invio.

Contenuto del ripetitore. annotazioni:

{'n': int, 'return': str, 's': str}

4 linguaggi di programmazione da imparare, anche se sei un umanista

L'era digitale detta le sue regole. Ora, la conoscenza dei linguaggi di programmazione si sta spostando dalla categoria delle "competenze troppo altamente specializzate" a quelle indispensabili per una carriera di successo. Abbiamo compilato per te i quattro linguaggi di programmazione più utili che ti aiuteranno a diventare uno specialista veramente efficace.

1. Vba

Se lavori costantemente in Excel e trascorri del tempo sulle stesse operazioni di routine ogni giorno, dovresti considerare di automatizzarle. Macro: le query nel linguaggio VBA integrato in Excel ti aiuteranno in questo. Avendo padroneggiato le macro, puoi non solo risparmiare tempo, ma anche liberare tempo per attività più interessanti (ad esempio, per scrivere nuove macro). A proposito, per automatizzare alcune operazioni, l'apprendimento di VBA non è necessario: basta registrare alcune azioni utilizzando [il registratore di macro](#), e puoi ripeterli di volta in volta. Ma è meglio imparare questa lingua: le registrazioni macro ti daranno solo funzionalità limitate.

Nonostante la sua attrattiva, le macro hanno due inconvenienti. Innanzitutto, poiché una macro esegue automaticamente tutte le azioni che faresti con le tue mani, carica seriamente la RAM (ad esempio, se hai 50.000 righe in un documento, il tuo povero computer potrebbe non resistere agli attacchi di macro). Il secondo: VBA come linguaggio di programmazione non è molto intuitivo e può essere difficile per un principiante impararlo da zero.

2. SQL

Per combinare dati provenienti da database diversi, è possibile utilizzare le formule VPR e INDICE (RICERCA). Ma cosa succede se hai dieci database diversi con 50 colonne di 10.000 righe ciascuna? È probabile che commettano un errore e passino troppo tempo.

Pertanto, è meglio utilizzare il programma Access. Come VPR, è progettato

per combinare dati da vari database, ma lo fa molto più velocemente. Access automatizza questa funzione, filtra rapidamente le informazioni e consente di lavorare in team. Oltre ad Access, esistono altri programmi per lavorare con i database (DBMS), nella maggior parte dei quali è possibile lavorare utilizzando un semplice linguaggio di programmazione: SQL (Structured Query Language). La conoscenza di SQL è uno dei requisiti di base per gli analisti aziendali insieme alla conoscenza di Excel, quindi ti consigliamo di dare un'occhiata più da vicino se vuoi lavorare in questo settore.

3. R

R è una buona alternativa a VBA se si desidera semplificare l'elaborazione dei dati. Come sta bene? Innanzitutto, a differenza di un collega, è davvero semplice e comprensibile anche per chi non ha mai avuto a che fare con la programmazione in vita sua. In secondo luogo, R è progettato per funzionare con i database e ha ampie funzionalità: può modificare la struttura di un documento, raccogliere dati da Internet, elaborare tipi di informazioni statisticamente differenti, costruire grafici, creare tag cloud, ecc. Per lavorare con file Excel, dovrai scaricare librerie speciali, ma è più conveniente salvare le tabelle da Excel in formato csv e lavorarci. A proposito, è R che è stato scritto il programma con cui facciamo TeamRoulette nei nostri campionati. Grazie a lei, riusciamo a disperdere le persone nelle squadre in cinque minuti invece di due ore.

Per imparare a lavorare in R, prima di tutto, scarica un ambiente di programmazione visivo e intuitivo - [RStudio](#).

4. Pitone

Python è un linguaggio di programmazione ancora più potente e popolare (a proposito, ha davvero [niente da fare](#) con pitone). Come R, Python ha [biblioteche speciali](#) che funzionano con i file Excel e sa anche come raccogliere informazioni da Internet (dimentica l'inserimento manuale dei dati nelle tabelle!). Puoi scrivere all'infinito su Python, ma se in poche parole è uno strumento davvero comodo e veloce che vale la pena padroneggiare se vuoi automatizzare le operazioni di routine, sviluppare il tuo pensiero algoritmico e in generale stare al passo con i progressi tecnici.

A differenza di R, Python non ha uno degli ambienti di programmazione più convenienti: qui ognuno è libero di scegliere di assaggiare. Ad esempio, noi

consiglia IDLE, Jupyter Notebook, Spyder, per i programmatori più avanzati - PyCharm.

Puoi imparare a programmare in Python e analizzare i dati utilizzandolo nel nostro corso online ["Pitone. Analisi dei dati"](#). I consulenti di Big4 e Big3 ti diranno come lavorare con le librerie, creare programmi e algoritmi per l'elaborazione delle informazioni.

Bene, per così dire, andiamo

Bene, se tu (You, He, She, It, They) stai leggendo questo significa che stiamo iniziando la nostra prima lezione di Python. Evviva (...) Oggi imparerai a programmare. E inoltre, imparerai a programmare in Python.

Ma per iniziare a programmare hai bisogno di Python. Dove trovarlo?



Dove trovare Python?

Ma pensi che ora puoi semplicemente sederti e merluzzo. Beh in realtà sì. MA di solito i veri hacker e programmatori codificano negli editor di codice. Per lavorare con Python, consigliamo di usare un Jupyter Notebook o JupyterLab o Visual Studio Code. (I) Codificheremo in Visual Studio Code, ma gli altri editor non sono diversi.

Bene, è tempo di CODIFICARE.

Sono troppo vecchio per questo

Tutti scrivono sempre il primo codice che mostra "Hello World". Bene,

siamo una specie di talento? No, e quindi cominceremo con questo. In Python, la funzione `print()` viene utilizzata per l'output del testo. Per produrre "Hello World" dobbiamo scrivere quanto segue:

```
1 print("Hello world!!!")
```

è vita

Sì, hai appena scritto il tuo primo programma. Ora sei un programmatore (NO).

Diamo un'occhiata alla struttura della funzione di stampa. Questa funzione consiste nel nome - `print` e nel luogo in cui gettiamo ciò che questa funzione dovrebbe produrre. Se vuoi visualizzare un testo, devi scriverlo tra virgolette, sia in singolo che in doppio.

Ma ora inseriamo del testo. Per questo, Python ha una funzione di input. Per accettare qualcosa, devi scrivere quanto segue:

```
1 a=input()  
2 print(a)
```

Cosa fa questo programma per noi? Per prima cosa introduciamo qualcosa, al momento non ci interessa cosa sia, testo o numeri. Dopo, il nostro programma mostra ciò che hai scritto.

Ma come essere sicuri di scrivere il testo prima di inserire qualcosa? Davvero semplice. Ci sono due modi per farlo.

```
1 print('      ' )  
2 a=input()  
3 print(a)
```

Come può, ma non è necessario

Se esegui il secondo programma noterai che devi inserire i dati nella stessa riga in cui è scritto il testo.

```
1 a=input('      ' )  
2 print(a)
```

Come fare

Ma questo non è esteticamente gradevole. Siamo qui con voi tutti gli esteti. Quindi, per poter inserire i valori su una riga separata, puoi fare quanto segue:

```
1 a=input( )
2 print(a)
```

Come bello, ma perché?

Come possiamo vedere, `\n` è apparso alla fine. Questo è un carattere di controllo che indica il passaggio alla riga successiva. Ma è troppo pigro per me scriverlo ogni volta, e quindi preferisco scrivere tutto nella riga in cui è scritto il testo.

Ma dove applicarlo? Molto semplice, puoi fare matematica. Ci sono alcune semplici operazioni aritmetiche in Python di cui ti parlerò ora.

Python ha le seguenti operazioni aritmetiche: addizione, sottrazione, moltiplicazione, divisione, divisione intera, elevazione a potenza e il resto della divisione. Poiché tutto è registrato, puoi vedere di seguito:

```
1 6+2
2 6-2
3 6*2
4 6/2
5 3//2
6 3**2
7 3%2
```

Finalmente imparerò a moltiplicare

Se vuoi stampare immediatamente il risultato, devi solo scrivere un'operazione aritmetica in stampa:

```
1 print(6+2)
```

Il risultato di questo programma sarà 8

Ma come si effettuano le operazioni con i numeri inseriti dall'utente? Dobbiamo usare l'input. Ma per impostazione predefinita in Python, i valori che l'input passa sono una stringa. E la maggior parte delle operazioni aritmetiche non può essere eseguita su una riga. Affinché il nostro programma funzioni come previsto, dobbiamo specificare quale tipo di valore deve utilizzare Python.

Per fare ciò, usiamo la funzione `int`:

```
1 a=int(input(": "))
2 b=int(input(": "))
3 print(a+b)
```

Sommiamo i due numeri che l'utente inserisce. (Wow, ho scoperto quanto sarà $2 + 2$)

1. OTTIMIZZAZIONE DEL CODICE PYTHON CON TIPI

Contenuto:

1. Ottimizzazioni di base
2. Stili
3. Compilazione Python
4. Strutture in Python
5. Chiama il tuo codice in C
6. Pypy

Ottimizzazioni di base

B Prima di riscrivere il codice sorgente Python in C, considerare i metodi di ottimizzazione di base in Python.

Strutture dati integrate

Le strutture dati integrate di Python, come set e dict, sono scritte in C. Funzionano molto più velocemente delle tue strutture dati composte come classi Python. Altre strutture dati oltre al set standard, dict, list e tuple sono descritte nella sezione [collezioni](#) documentazione del modulo.

Elenco espressioni

Invece di aggiungere elementi all'elenco utilizzando il metodo standard, utilizza le espressioni di elenco.

```
#Slow
mapped = []
for value in originallist:
    mapped.append(myfunc(value))

# Faster
mapped = [myfunc(value) in originallist]
```

tipi di c

Il [tipi di c](#) module ti consente di interagire con il codice C di Python senza utilizzare un sottoprocessore del modulo un altro modulo simile per avviare altri processi dalla CLI.

Ci sono solo due parti: compilare il codice C da caricare in oggetti condivisi di qualità e impostare le strutture dati nel codice Python per mapparle ai tipi C.

In questo articolo, combinerò il mio codice Python con [LCS.c](#), che [trova](#) la sottosequenza più lunga negli elenchi di due righe. Voglio che quanto segue funzioni in Python:

```
list1 = ['My', 'name', 'is', 'Sam', 'Stevens', '!']
list2 = ['My', 'name', 'is', 'Alex', 'Stevens', '.']

common = lcs(list1, list2)

print(common)
# ['My', 'name', 'is', 'Stevens']
```

Un problema è che questa particolare funzione C è la firma di una funzione che accetta elenchi di stringhe come tipi di argomenti e restituisce un tipo che non ha una lunghezza fissa. Risolvo questo problema con una struttura di sequenza contenente puntatori e lunghezze.

Compilare codice C in Python

Innanzitutto, il codice sorgente C (`lcs.c`) viene compilato in `lcs.so` per essere caricato in Python.

```
gcc -c -Wall -Werror -fpic -O3 lcs.c -o lcs.o
gcc -shared -o lcs.so lcs.o
```

- Wall visualizzerà tutti gli avvisi:
- L'errore avvolgerà tutti gli avvisi in errori:

- fpic genererà istruzioni indipendenti dalla posizione di cui avrai bisogno se vuoi usare questa libreria in Python:
- O3 massimizza l'ottimizzazione:

E ora inizieremo a scrivere codice Python utilizzando il file oggetto condiviso risultante.

Strutture in Python

Di seguito sono riportate due strutture di dati utilizzate nel mio codice C.

```
struct sequence
{
    char ** items:
    int length:
}:

struct cell
{
    int index:
    int length:
    struct Cell * prev:
}:
```

Ed ecco la traduzione di queste strutture in Python.

```

import ctypes
class SEQUENCE (ctypes.Structure):
    _fields_ = [('items', ctypes.POINTER
(ctypes.c_char_p)),
                ('length', ctypes.c_int)]

class CELL (ctypes.Structure):
    pass

CELL._fields_ = [('index', ctypes.c_int),
('length', ctypes.c_int),
                ('prev', ctypes.POINTER (CELL))]

```

Alcune note:

- Tutte le strutture sono classi che ereditano da ctypes.Structure.
- L'unico campo `_fields_` è un elenco di tuple. Ogni tupla è (<nome-variabile>, <ctypes.TYPE>).
- Ci sono tipi simili in `c_char` (char) e `c_char_p` (* char) .
- C'è anche ctypes un `POINTER()` che crea un puntatore di tipo da ogni tipo passato ad esso.
- Se hai una definizione ricorsiva come in CELL, devi passare la dichiarazione iniziale e quindi aggiungere i campi `_fields_` per ottenere un collegamento a te stesso in seguito.
- Dato che non ho usato CELLPython nel mio codice, non ho avuto bisogno di scrivere questa struttura, ma ha una proprietà interessante nel campo ricorsivo.

Chiama il tuo codice in C

Inoltre, avevo bisogno di codice per convertire i tipi Python in nuove strutture in C. Ora puoi usare la tua nuova funzione C per velocizzare il codice Python.

```

def list_to_SEQUENCE (strlist: List [str]) ->
SEQUENCE:
    bytelist = [bytes (s, 'utf-8') for s in strlist]
    arr = (ctypes.c_char_p * len (bytelist)) ()
    arr [:] = bytelist
    return SEQUENCE (arr, len (bytelist))

def lcs (s1: List [str], s2: List [str]) -> List [str]:
    seq1 = list_to_SEQUENCE (s1)
    seq2 = list_to_SEQUENCE (s2)

# struct Sequence * lcs (struct Sequence * s1,
struct Sequence * s2)
common = lcsmodule.lcs (ctypes.byref (seq1),
ctypes.byref (seq2)) [0]

    ret = []

    for i in range (common.length):
        ret.append (common.items [i] .decode ('utf-
8')) lcsmodule.freeSequence (common)

    return ret

lcsmodule = ctypes.cdll.LoadLibrary ('lcsmodule
/ lcs.so')
lcsmodule.lcs.restype = ctypes.POINTER
(SEQUENCE)

list1 = ['My', 'name', 'is', 'Sam', 'Stevens', '!']
list2 = ['My', 'name', 'is', 'Alex', 'Stevens', '.']

common = lcs (list1, list2)

print (common)

```

Alcune note:

- `** char` (elenco di stringhe) corrisponde direttamente a un elenco di byte in Python.
- C'è `lcs.cis` una funzione `lcs()` con la firma `struct Sequence * lcs(struct Sequence * s1, struct Sequence`
- `s2)` . Per impostare il tipo restituito, utilizzo `lcsmodule.lcs.restype = ctypes.POINTER(SEQUENCE)`.
- Per effettuare una chiamata con un riferimento alla struttura `Sequence`, utilizzo `ctypes.byref()` che restituisce un "puntatore luminoso" al tuo oggetto (più veloce di `ctypes.POINTER()`).
- `common.items`- questo è un elenco di byte, possono essere decodificati per ottenere la forma di un elenco `str`.
- `lcsmodul.freeSequence` (`common`) libera solo la memoria associata a `common`. Questo è importante perché il Garbage Collector (AFAIK) non lo raccoglierà automaticamente.

Il codice Python ottimizzato è codice che hai scritto in C e racchiuso in Python.

Qualcosa di più: PyPy

Attenzione: io stesso non ho mai usato PyPy.

Una delle ottimizzazioni più semplici è eseguire i programmi nel [PyPy](#) runtime, che contiene un compilatore JIT (just-in-time) che velocizza il lavoro dei cicli, compilandoli in codice macchina per l'esecuzione ripetuta.

2. TROVARE IL TOOLKIT PERFETTO: ANALIZZARE IL POPOLARE PYTHON MODELLI DI PROGETTO

TI materiali, di cui pubblichiamo oggi la traduzione, sono dedicati alla storia degli strumenti utilizzati per creare applicazioni Python. È progettato per quei programmatori che hanno già lasciato la categoria dei principianti ma non hanno ancora raggiunto la categoria degli sviluppatori Python esperti.

Per coloro che non vedono l'ora di iniziare la pratica, l'autore suggerisce di utilizzare [Flake8](#), [pytest](#), e [Sfinge](#) nei progetti Python esistenti. Consiglia anche di dare un'occhiata [pre-commit](#), [Nero](#), e [Pylint](#). Coloro che hanno intenzione di iniziare un nuovo progetto, consiglia di prestare attenzione a [Poesia](#) e [Affidabile](#).

Panoramica

È sempre stato difficile per me formarmi un'opinione obiettiva sulle "migliori pratiche" dello sviluppo di Python. Nel mondo della tecnologia, emergono continuamente alcune tendenze popolari, che spesso non durano a lungo. Ciò complica l'estrazione del "segnale utile" dal rumore informativo.

Gli strumenti più recenti sono spesso buoni, per così dire, solo sulla carta. Possono aiutare il programmatore pratico con qualcosa? Oppure la loro applicazione porta solo all'introduzione di qualcosa di nuovo nel progetto, le cui prestazioni devono essere mantenute, che comporta più difficoltà che benefici?

Non avevo una chiara comprensione di cosa considerassi esattamente le "migliori pratiche" di sviluppo. Suppongo di aver trovato qualcosa di utile, basato principalmente su prove episodiche di "utilità" e sulla menzione occasionale di ciò nelle conversazioni. Ho deciso di mettere le cose in ordine in questa faccenda. Per fare questo, ho iniziato ad analizzare tutti i template dei progetti Python che riuscivo a trovare (stiamo parlando dei template usati [dal tagliabiscotti](#) riga di comando [utilità](#) per creare progetti Python basati su di essi).

Mi è sembrato affascinante apprendere quali strumenti ausiliari gli autori dei modelli considerano degni di inserire questi strumenti in nuovi progetti Python creati sulla base di questi modelli.

Ho analizzato e confrontato i 18 progetti di template più popolari (da 76 a 6300 stelle su GitHub), prestando particolare attenzione al tipo di strumenti ausiliari che utilizzano. I risultati di questo esame possono essere trovati in [questo tavolo](#).

Di seguito voglio condividere le principali conclusioni che ho tratto durante l'analisi di modelli popolari.

Standard di fatto

Gli strumenti discussi in questa sezione sono inclusi in più della metà dei modelli. Ciò significa che sono percepiti come standard in un gran numero di progetti Python reali.

fiocco8

ho usato [fiocco8](#) da [un po'](#) di tempo, ma non sapevo della posizione dominante nel campo del pelucchi che questo strumento occupa. [Essopensava che esistesse](#) in una sorta di competizione, ma la stragrande maggioranza dei modelli di progetto lo usa.

Sì, questo non è sorprendente. È difficile opporre qualcosa alla comodità di linting dell'intera base di codice del progetto in una manciata di secondi. Coloro che desiderano utilizzare uno sviluppo all'avanguardia possono consigliare uno sguardo [a make-python-styleguide](#). È qualcosa come "Flake8 con steroidi". Questo strumento potrebbe ben contribuire al trasferimento nella categoria degli obsoleti di altri strumenti simili (come Pylint).

Pytest e coverage.py

La stragrande maggioranza dei modelli [usa pytest](#). Questo riduce l'uso del quadro standard di unit test. Pytest sembra ancora più attraente se combinato con [tossico](#). Questo è [esattamente](#) ciò che è stato fatto in circa la metà dei modelli. La copertura del codice con i test viene spesso verificata utilizzando [copertura.py](#).

Sfinge

La maggior parte dei modelli [usa Sfinge](#) per generare documentazione. Con mia sorpresa, [MkDocs è raramente](#) utilizzato per questo scopo.

Di conseguenza, possiamo dire che se non usi Flake8, pytest e Sphinx nel tuo progetto attuale, dovresti considerare di introdurli.

Strumenti promettenti

In questa sezione ho raccolto quegli strumenti e quelle tecniche, il cui uso nei modelli ha suggerito alcune tendenze. Il punto è che, sebbene tutto questo non appaia nella maggior parte dei modelli di progetto, si trova in molti modelli relativamente recenti. Quindi, vale la pena prestare attenzione a tutto questo.

Pyproject.toml

L'utilizzo del file è pyproject.toml suggerito in [PEP 518](#). Questo è un meccanismo moderno per specificare i requisiti di assemblaggio del progetto. È usato nella maggior parte dei modelli abbastanza giovani.

Poesia

Sebbene l'ecosistema Python non stia andando bene in termini di strumento eccellente per la gestione delle dipendenze, sono cautamente ottimista sul fatto che [Poesia](#) potrebbe essere l'equivalente di npm dal mondo JavaScript nel mondo Python.

I modelli di progetto più giovani (ma popolari) sembrano concordare con questa mia idea. È vero, vale la pena dire che se qualcuno sta lavorando a una sorta di libreria che può pianificare di distribuire attraverso [PyPI](#), poi dovrà ancora usare [strumenti di configurazione](#). (Va notato che dopo la pubblicazione di questo materiale, sono stato informato che questo non è più un problema).

Inoltre, fai attenzione se il tuo progetto (lo stesso vale per le dipendenze) si basa su [Conda](#). In questo caso, Poetry non ti si addice, poiché questo strumento, nella sua forma attuale, vincola lo sviluppatore a [pip](#) e [virtualenv](#).

Dipendentebot

[Dipendente](#) controlla regolarmente le dipendenze del progetto per l'obsolescenza e cerca di aiutare lo sviluppatore aprendo automaticamente PR.

Di recente ho visto questo strumento più spesso di prima. Mi sembra che sia un ottimo strumento: la cui aggiunta al progetto incide molto positivamente sul progetto. Dependabot aiuta a ridurre i rischi per la sicurezza spingendo gli sviluppatori a mantenere aggiornate le dipendenze.

Di conseguenza, ti ho consigliato di non perdere di vista Poetry e Dependabot. Considera l'introduzione di questi strumenti nel tuo prossimo progetto.

Consigli personali

L'analisi dei modelli di progetto mi ha dato una percezione un po' ambivalente degli strumenti che elencherò in questa sezione. In ogni caso, voglio usare questo

sezione per raccontarli, in base alla mia esperienza. Un tempo mi erano utili.

Pre-impegno

Anche se sei incredibilmente disciplinato, non sprecare le tue energie nell'esecuzione di semplici azioni di routine come il codice aggiuntivo eseguito attraverso il linter prima di inviare il codice al repository. Compiti simili possono essere passati a [Pre-commettere](#). Ed è meglio spendere le tue energie su TDD e il lavoro di squadra sul codice.

Pylint

Sebbene [Pylint è criticato](#) per essere lento, sebbene questo strumento sia criticato per le caratteristiche delle sue impostazioni, posso dire che mi ha permesso di crescere al di sopra di me stesso nel campo della programmazione.

Mi dà istruzioni specifiche su quelle parti del codice che posso migliorare, mi dice come rendere il codice più conforme alle regole. Per uno strumento gratuito, questo da solo è già molto. Pertanto, sono pronto a sopportare l'inconveniente associato a Pylint.

Nero

Il nero alla radice del dibattito su dove inserire gli spazi nel codice. Questo protegge i nostri team da chiacchiere vuote e differenze insignificanti nei file causate dalle diverse impostazioni degli editor.

Nel mio caso, illumina ciò che non mi piace di Python (la necessità di utilizzare molti spazi). Inoltre, va notato che il progetto Black nel 2019 ha aderito alla Python Software Foundation, il che indica la serietà e la qualità di questo progetto.

Di conseguenza, voglio dire che se ancora non usi pre-commit, Black e Pylint, pensa se questi strumenti possono avvantaggiare la tua squadra.

Subtotali

Dodici dei diciotto modelli esaminati sono stati creati utilizzando il

[tagliabiscotti](#) struttura. Alcuni di quei modelli in cui questo framework non viene utilizzato hanno qualità entusiasmanti.

Ma dato che cookiecutter è il framework principale per la creazione di modelli di progetto, coloro che decidono di utilizzare un modello che non utilizza un cookiecutter dovrebbero avere ottime ragioni per questo. Tale decisione dovrebbe essere molto ben giustificata.

Coloro che sono alla ricerca di un modello per il proprio progetto dovrebbero scegliere un modello che si avvicina di più alla sua visione delle cose. Se tu, quando crei progetti secondo un modello preciso, devi continuamente riconfigurarli allo stesso modo, pensa a come creare un tale modello e perfezionarlo, ispirandoti agli esempi di modelli della mia lista.

E se sei attratto dall'avventura - [creare il tuo modello](#) da zero. Cookiecutter è un'eccellente funzionalità dell'ecosistema Python e il semplice processo di [creazione Modelli Jinja](#) ti consente di fare qualcosa di tuo in modo rapido e semplice.

Bonus: consigli sui modelli

Django

Insieme ai modelli Django più popolari, considera l'utilizzo [facciamo-__django-modello](#). Dà l'impressione di un prodotto profondamente pensato.

Elaborazione e analisi dei dati

Nella maggior parte dei progetti volti all'elaborazione e all'analisi dei dati, il [Cookiecutter Data Science](#) il modello è utile. Tuttavia, gli scienziati dei dati dovrebbero anche guardare [Kedro](#).

Questo modello estende Cookiecutter Data Science con un meccanismo per la creazione di pipeline di elaborazione dati standardizzate. Supporta il caricamento e il salvataggio di dati e modelli. È molto probabile che queste funzionalità siano in grado di trovare un'applicazione degna nel tuo prossimo progetto.

Qui vorrei anche esprimere la mia gratitudine ai creatori del [shablona](#) modello per la preparazione di documentazione di altissima qualità. Può esserti utile anche se alla fine scegli qualcos'altro.

Modelli per scopi generici

Quale modello generico scegliere in qualche modo dipende da cosa esattamente svilupperai in base a questo modello: una libreria o una normale applicazione. Ma io, selezionando un modello simile, insieme ai progetti più popolari di questo tipo, guarderei molto da vicino [Modello Python di Jace](#).

Questo non è uno schema ben noto, ma mi piace il fatto che abbia Poesia, [isort](#), Nero, pylint e [mypy](#).

[PyScaffold](#) è uno dei modelli più popolari non basati su cookiecutter. Ha molte estensioni (ad esempio, per Django e [Progetti di Data Science](#)). Scarica anche i numeri di versione da GitHub usando [setuptools-scm](#). Inoltre, questo è uno dei pochi modelli che supportano Conda.

Ecco un paio di modelli che utilizzano la tecnologia GitHub Actions:

1. Il [Modello di tagliabiscotti delle migliori pratiche di Python](#), che, voglio sottolineare, ha la maggior parte dei miei strumenti preferiti.
2. Il [Blueprint / Boilerplate per progetti Python template](#), che trovo piuttosto interessante, come la capacità che dà loro di trovare problemi di sicurezza comuni con [Bandito](#), sembra promettente. Inoltre, questo modello ha una caratteristica notevole, che consiste nel fatto che le impostazioni di tutti gli strumenti sono raccolte in un unico file setup.cfg.

E infine - mi raccomando [prendendo un](#) guarda a [realizziamo-pacchetto-python](#) modello. Penso che valga la pena farlo comunque. In particolare, se ti piace il modello Django dello stesso sviluppatore, o se utilizzerai l'avanzato, [facciamo-python-styleguide](#) invece del puro Flake8.

3. COME VENGONO IMPLEMENTATI I TIPI DI INTERI AMPIO IN PYTHON?

W Quando scrivi in un linguaggio di basso livello come il C, sei preoccupato di scegliere il tipo di dati e i qualificatori giusti per i tuoi interi, ad ogni passaggio, analizzi se sarà sufficiente utilizzarlo semplicemente o se è necessario aggiungere long o anche lungo doppio. Tuttavia, quando scrivi codice in Python, non devi preoccuparti di queste cose "minori", perché Python può funzionare con numeri di tipo intero di dimensione.

```
#include <stdio.h>
#include <math.h>

int main (void) {
    printf ("%Lf\n", powl (2, 20000));
    return 0;
}
```

```
$ ./a.out
inf
```

In C, se

provi a calcolare 220.000 utilizzando la funzione integrata powl, otterrai il uscita inf.

```
>>> 2 ** 20,000
39802768403379665923543072061912024537047727804924259387134 ...
...
... 6021 digits long ...
...
6309376
```

Ma in

Python, renderlo più semplice che mai è facile:

Deve essere nascosto che Python sta facendo qualcosa di molto bello, e oggi scopriremo esattamente cosa fa per lavorare con numeri interi di dimensioni arbitrarie!

Presentazione e definizione

Intero in Python, questa è una struttura C definita come segue:

```
struct _longobject {
    PyObject_VAR_HEAD
    digit ob_digit[1];
};
```

PyObject_VAR_HEAD is a macro, it expands to PyVarObject, which has the following structure:

```
typedef struct {
    PyObject ob_base;
    Py_ssize_t ob_size: /* Number of items in variable part */
} PyVarObject;
```

Altri tipi che hanno PyObject_VAR_HEAD:

- PyBytesObject
- PyTupleObject
- PyListObject

Ciò significa che un intero, come una tupla o una lista, ha una lunghezza variabile, e questo è il primo passo per capire come Python può supportare il lavoro con i numeri giganti. Una volta espansa, la macro _longobject può essere considerata come:

```
struct _longobject {
    PyObject ob_base;
    Py_ssize_t ob_size: /* Number of items in variable part */
    digit ob_digit[1];
};
```

Ci sono PyObject alcuni meta campi nella struttura che servono per il conteggio dei riferimenti (garbage collection), ma per parlare di questo serve un articolo a parte. Il campo su cui concentreremo questo ob_digit and in un bit ob_size.

Decodifica ob_digit

ob_digit È un array allocato staticamente di lunghezza unitaria di tipo digit (typedef для uint32_t). Poiché questo è un array, ob_digit is è un puntatore principalmente a un numero e quindi, se necessario, può essere aumentato utilizzando la funzione malloc a qualsiasi lunghezza. In questo modo, Python può rappresentare ed elaborare numeri molto grandi.

In genere, nei linguaggi di basso livello come il C, la precisione degli interi è limitata a 64 bit. Tuttavia, Python supporta interi di precisione arbitraria. A partire da Python 3, tutti i numeri sono presentati nella forma bignum e sono limitati solo dalla memoria di sistema disponibile.

Decodifica ob_size

ob_size memorizza il numero di elementi in ob_digit. Python sovrascrive e quindi usa il valore ob_size per determinare il numero effettivo di elementi contenuti nell'array per aumentare l'efficienza dell'allocazione della memoria all'array ob_digit.

Magazzinaggio

Il modo più ingenuo per memorizzare numeri interi è memorizzare una cifra decimale in un elemento dell'array. Operazioni come addizioni e sottrazioni possono essere eseguite secondo le regole della matematica della scuola elementare.

Con questo approccio, il numero 5238 verrà salvato in questo modo:

Questo approccio è inefficiente perché utilizzeremo cifre fino a 32 bit (uint32_t) per memorizzare una cifra decimale, che va da 0 a 9 e può essere facilmente rappresentata con solo 4 bit. Dopotutto, quando si scrive qualcosa di universale come Python, lo sviluppatore del kernel deve essere ancora più creativo.

Allora, possiamo fare di meglio? Ovviamente, altrimenti, non avremmo pubblicato questo articolo. Diamo un'occhiata più da vicino a come Python memorizza un intero extra lungo.

Percorso Python

Invece di memorizzare solo una cifra decimale in ogni elemento dell'array ob_digit, Python converte i numeri del sistema numerico con base 10 nei numeri del sistema con base 230 e chiama ogni elemento come un numero il cui valore varia da 0 a 230 - 1.

Nel sistema numerico esadecimale, la base 16 ~ 24 significa che ogni "cifra" del numero esadecimale varia da 0 a 15 nel sistema numerico decimale. In Python, è simile a un "numero" con base 230, il che significa che il numero andrà da 0 a 230 - 1 = 1073741823 in decimale.

In questo modo, Python utilizza efficacemente quasi tutto lo spazio allocato di 32 bit per cifra, risparmia risorse ed esegue ancora operazioni semplici, come addizioni e sottrazioni a livello di matematica della scuola elementare.

A seconda della piattaforma, Python utilizza array di interi senza segno a 32 bit o array di interi senza segno a 16 bit con cifre a 15 bit. Per eseguire le operazioni che verranno discusse in seguito, occorrono solo pochi bit.

Esempio: 1152921504606846976

Come già accennato, per Python i numeri sono rappresentati in un sistema con base 230, ovvero se converti 115291504606846976 in un sistema numerico con base 230, otterrai 100.

$$1152\ 9215\ 0460\ 6846\ 976 = 1 * ((230)^2 + 0) * ((230)^1 + 0) * ((230)^0)$$

Poiché è `ob_digitfirst` a memorizzare la cifra meno significativa, viene memorizzata come 001 in tre cifre. La struttura `_longobject` per questo valore conterrà:

- `ob_size` come 3
- `ob_digit` come [0, 0, 1]

Abbiamo creato una [demo REPL](#) che mostrerà come Python memorizza un intero al suo interno e si riferisce anche a membri strutturali come `ob_size`, `ob_refcount` etc.

Operazioni lunghe intere

Ora che abbiamo un'idea chiara di come Python implementa interi di precisione arbitraria, è tempo di capire come vengono eseguite varie operazioni matematiche con essi.

aggiunta

Gli interi sono memorizzati "in numeri", il che significa che l'addizione è semplice come nella scuola elementare, e il codice sorgente Python ci mostra che è così che viene implementata l'addizione. Una funzione con un nome `x_add` in un file [longobject.c](#) aggiunge due numeri.

```
...
For (i = 0; i < size_b; ++i) {
    carry += a->ob_digit[i] + b->ob_digit[i];
    z->ob_digit[i] = carry & PyLong_MASK;
    carry >>= PyLong_SHIFT;
}
For (i = 0; i < size_a; ++i) {
    carry += a->ob_digit[i];
    z->ob_digit[i] = carry & PyLong_MASK;
    carry >>= PyLong_SHIFT;
}
z->ob_digit[i] = carry;
...
```

Il frammento di codice sopra è preso da una funzione `x_add`. Come puoi vedere, itera su un numero per numero ed esegue l'aggiunta di numeri, calcola il risultato e aggiunge la sillabazione.

Diventa più interessante quando il risultato dell'addizione è un numero negativo. Il segno `ob_size` è un segno intero, cioè se hai un segno negativo

numero, allora `ob_size` sarà un meno. Il valore `ob_size` modulo determinerà il numero di cifre in `ob_digit`.

Sottrazione

Così come avviene l'addizione, avviene anche la sottrazione. Una funzione con un `nomex_sub` nel file [longobject.c](#) sottrae un numero da un altro.

```
...
For (i = 0; i < size_b; ++i) {
    borrow = a->ob_digit[i] - b->ob_digit[i] - borrow;
    z->ob_digit[i] = borrow & PyLong_MASK;
    borrow >>= PyLong_SHIFT;
    borrow &= 1; /* Keep only one sign bit */
}
for (; i < size_a; ++i) {
    borrow = a->ob_digit[i] - borrow;
    z->ob_digit[i] = borrow & PyLong_MASK;
    borrow >>= PyLong_SHIFT;
    borrow &= 1; /* Keep only one sign bit */
}
...
```

Il frammento di codice sopra è preso da una funzione `x_sub`. In esso, puoi vedere come avviene l'enumerazione dei numeri e la sottrazione, il risultato viene calcolato e il trasferimento viene distribuito. In effetti, è molto simile all'addizione.

Moltiplicazione

E ancora, la moltiplicazione sarà attuata nello stesso modo ingenuo che abbiamo imparato dalle lezioni di matematica alle elementari, ma non è molto efficiente. Per mantenere l'efficienza, Python implementa l'[algoritmo di Karatsuba](#), che moltiplica due numeri di n cifre in semplici passaggi $O(n \log 23)$.

L'algoritmo non è semplice e la sua implementazione esula dallo scopo di questo articolo, ma puoi trovare la sua implementazione nelle funzioni e nel file [.k_mul_k_lopsided_mullongobject.c](#)

Divisione e altre operazioni

Tutte le operazioni sugli interi sono definite nel file [longobject.c](#), sono molto semplice trovare e rintracciare il lavoro di ciascuno. Attenzione: una comprensione dettagliata del lavoro di ciascuno di essi richiederà tempo, quindi fai scorta di popcorn.

Ottimizzazione dei numeri interi usati di frequente

Python prealloca un piccolo numero di interi in memoria che vanno da -5

a 256. Questa allocazione si verifica durante l'inizializzazione, e poiché non possiamo cambiare i numeri interi (immutabilità), questi numeri pre-allocati sono singleton e sono referenziati direttamente invece di essere allocati. Ciò significa che ogni volta che usiamo/creiamo un numero piccolo, Python invece della riallocazione restituisce semplicemente un riferimento al numero precedentemente allocato.

Tale ottimizzazione può essere rintracciata nella macro `IS_SMALL_INT` e funzione [get_small_int](#) in `longobject.c`. Quindi Python risparmia molto spazio e tempo nel calcolo dei numeri interi comunemente usati.

4. CREA UN BOT IN PYTHON PER IMPARARE INGLESE

no, questo non è uno delle centinaia di articoli su come scrivere il tuo primo bot Hello World in Python. Qui non troverai istruzioni dettagliate su come ottenere un token API in BotFather o avviare un bot nel cloud. In cambio, ti mostreremo come liberare tutta la potenza di Python al massimo per ottenere il massimo dell'estetica e della bellezza

codice. Eseguiamo una canzone sul fascino di strutture complesse: balliamo e balliamo. Sotto l'asincronia tagliata, il suo sistema di salvataggi, un mucchio di utili decoratori e un sacco di bel codice.

Disclaimer: Le persone con OOP cerebrale e aderenti ai modelli "giusti" possono ignorare questo articolo.

Idea

Per capire cosa vuol dire non conoscere l'inglese nella società moderna, immagina di essere un nobile del XVIII secolo che non conosce il francese. Anche se non sei molto esperto di storia, puoi ancora immaginare quanto sarebbe difficile vivere in tali circostanze. Nel mondo moderno, l'inglese è diventato una necessità, non un privilegio, soprattutto se sei nel settore IT.

Il progetto si basa sul catechismo del futuro: lo sviluppo di una rete neurale come unità separata e l'educazione, che si basa sul gioco e sullo spirito sportivo. I paradigmi isomorfici sono sospesi nell'aria fin dall'antichità, ma sembra che nel tempo le persone abbiano cominciato a dimenticare che le soluzioni più semplici sono le più efficaci.

Ecco una breve lista delle cose di base che voglio mettere insieme:

- Essere in grado di lavorare con tre dizionari utente
- Possibilità di analizzare video/testo di YouTube e quindi aggiungere nuove parole al dizionario dell'utente
- Due modalità di allenamento delle abilità di base

- Personalizzazione flessibile: pieno controllo sui dizionari utente e sull'ambiente in generale
- Pannello di amministrazione integrato

Naturalmente, tutto dovrebbe funzionare rapidamente, con la possibilità di ripristinare facilmente le funzionalità esistenti in futuro. Mettendo tutto insieme, ho pensato che la migliore incarnazione della mia idea nella realtà sarebbe stata un bot di Telegram. La mia storia non riguarda come scrivere correttamente i gestori per il bot: ci sono dozzine di tali articoli e questo è un semplice lavoro meccanico. Voglio che il lettore impari a ignorare i dogmi tipici della programmazione. Usa ciò che è redditizio ed efficace qui e ora.

"Ho imparato a far uscire le grida degli increduli dalle mie orecchie perché era impossibile reprimerle".

Struttura di base

Il bot si baserà sul [python-telegram-bot \(ptb\)](#) biblioteca. Io uso [loguru](#) come un logger, anche se c'è un piccolo intoppo qui. Il fatto è che ptb per impostazione predefinita utilizza un logger diverso (registrazione standard) e non ti consente di connetterti da solo. Certo, sarebbe possibile intercettare tutti i messaggi del journal a livello globale e inviarli al nostro registrar, ma lo faremo in modo un po' più semplice:

```
from loguru import logger
import sys

# Configure dual-stream output: to the console and to the file
config = {
    'handlers': [
        {'sink': sys.stdout, 'level': 'INFO'},
        {'sink': 'logs.log', 'serialize': False, 'level': 'DEBUG'},
    ],
}

logger.configure(**config)

# ...

updater = Updater('YOUR_TOKEN')
dp = updater.dispatcher

# Roughly fasten your logger. Cheap and cheerful
updater.logger = logger
dp.logger = logger
```

Sfortunatamente, non ho l'opportunità di distribuire il mio bot su stable

data center, quindi la sicurezza dei dati è una priorità. Per questi scopi, ho implementato il mio sistema di salvataggi. Fornisce un lavoro flessibile e conveniente con la raccolta di dati e statistiche, come esempio di un facile rifornimento di funzionalità in futuro.

```
from __future__ import annotations # In the future, I will omit this import
from loguru import logger

import inspect
import functools
import os

def file_is_empty(path: str) -> bool:
    return os.stat(path).st_size == 0

def clear_file(path: str) -> None:
    with open(path, 'w'): pass

def cache_decorator(method):
    @functools.wraps(method)
    def wrapper(self, *args, **kwargs):
        res = method(self, *args, **kwargs)
        Cache.link.recess(self, {'method_name': method.__name__}) # (1)
        # During testing, multiple calls can slow down
        # program work: opt avoids this
        logger.opt(lazy = True).debug(f'Decorator for {method.__name__} was end ')
        return res
    return wrapper

class cache:
    """
```

+ cache_size - Std thrDtlgh 'hi ch dl data will be su'ed

+ cache_files - File di dump in cui sono tutte le operazioni intermedie sui dati

immagazzinato

collegamento = Nessuno

def dentro (self, cache_size = 10):

= Salva Tutti avvitato cl asini. questo S Tutti o's tu T o fl esibibile' lavoro insieme a dati.

sel F. clasini = []

=File classi corrispondenti

sel f._cache_files = []

= (1). UN cmq dl mod Quello dl o's tu a cd1 a istanza specifica thrDtlgh un comune classe

= questo S w'wks perché noi oril \ ' avere uno esempio Df il classe. 'Ciao ch

= attrezzi d 1 il logica di 'orking insieme a dati. Inoltre Su. è

conveniente ari D Tutti o 's

= ns gomma cantl>' espandere il funzione Dflallt>' in il futuro

se stesso. classe .collegamento = sé

self._counter = 0

self.CACHE DIMENSIONE = cache ns ze

def add (auto, D S: classe, file: str) -> NDne:

Tutto o è per allacciare un classe a un sa vero

+ cls - Installa un ce di il classe

+ file - Il fi1 e il instari ce è 'operare' insieme a

self._cache_files.append (file)

self._cl asses.append (cl)

se fi1e i S eotptl' (fi1e): Restituzione hDne

1 ogger. optare D un.' = True) .debug (fTo {cl S .
file (file) ènon vuoto' ')

nomi delle classi)

fa dati in vende oad (file):

è.salva nDn Caclti ng (dati)

cl ear_file (file)

sel f._counter = 0

def recesso (se stesso, cl: ct culo, dati dieta) —> Ncri e:

Il principale metodo Quello esegue il basi cl o c di salva

Se se stesso. contatore + 1/ = self.CACHE DIMENSIONE:

auto.salvataggio Tutti 0

el si:

self._counter += io

file nome = self._cache_files [self._classes.index (cls)]

auto.salvataggio (dati. file enam e = nome del file)

= Per simplicit>', save_d1, salva, carica metodi sono omesso

Ora possiamo creare qualsiasi metodo in grado di modificare i dati, senza timore di

perdere dati importanti:

```
@cache_decorator
def add_smth_important (* args, ** kwargs) -> Any:
    # ...
    # We make some important actions on the data ...
    # ...
```

Ora che abbiamo capito la struttura di base, rimane la domanda principale: come mettere insieme il tutto. Ho implementato la classe principale - EnglishBot, che riunisce l'intera struttura primaria: PTB, lavora con il database, il sistema di salvataggio e che gestirà l'intera logica di business del bot. Se l'implementazione dei comandi di Telegram fosse semplice, potremmo facilmente aggiungerli alla stessa classe. Ma, sfortunatamente, la loro organizzazione occupa la maggior parte del codice dell'intera applicazione, quindi aggiungerli alla stessa classe sarebbe da pazzi. Inoltre non volevo creare nuove classi/sottoclassi, perché suggerisco di utilizzare una struttura molto semplice:

```
# Import the main class
from modules import EnglishBot
# Import classes that implement Telegram commands
from modules.module import start
# ...

if __name__ == '__main__':
    # Initialize the bot
    tbot = EnglishBot (
        # ...
    )

    # Add handlers to the stack
    tbot.add_command_handler (start, 'start')
    # ...
```

Considereremo ulteriormente il modo in cui i moduli di comando ottengono l'accesso alla classe principale.

Tutto dal nulla

I gestori Ptb hanno due argomenti: aggiornamento e contesto, che memorizzano tutto lo stack di informazioni necessario. Il contesto ha un meraviglioso argomento chat_data che può essere utilizzato come dizionario di archiviazione dati per la chat. Ma non voglio farvi riferimento costantemente in un formato context.chat_data['data']. Vorrei qualcosa di leggero e bello, diciamo context.data. Tuttavia, questo non è un problema.

```

from telegram.ext import CommandHandler

def a(self, key: str):
    # First, check if the class has the required value
    # If not, then try to return it from chat_data
    try:
        return object.__getattribute__(self, key)
    except:
        return self.chat_data[key]

def b(self, key: str, data = None, replace = True):
    # Small hack: if replace = False and data exists, then overwriting does not occur
    # In this case, if the data is not specified, then they are put in None
    if replace or not self.chat_data.get(key, None):
        self.chat_data[key] = data

# Bindin context to receive data in the format context.data
CallbackContext.__getattribute__ = a
# As well as a convenient setter for your needs
CallbackContext.set = b

```

Continuiamo a semplificare la nostra vita. Ora voglio che tutte le informazioni necessarie per un utente specifico siano in un contesto di accesso rapido.

```

def bind_context(func):
    def wrapper(update, context):
        context._bot.bind_user_data(update, context) # (2)
        return func(update, context)
    return wrapper

class EnglishBot:
    # ...

    def bind_user_data(self, update, context) -> dict:
        context.set('t_id', update.message.chat_id, replace = False)
        context.set('t_ln', update.message.from_user.language_code, replace = False)
        # ...
        # Set all the necessary information that we want to have quick access from context
        # For example, something from the database
        # ...

```

Ora diventeremo completamente impudenti e fisseremo la nostra istanza del bot al contesto:

```

class EnglishBot:
    # ...

    def __init__(self, * args, ** kwargs):
        # ...
        # (2): Now we can access the instance in the format context._bot
        CallbackContext._bot = self

```

Mettiamo tutto a posto e otteniamo una cittadella di comfort e convenienza in una sola chiamata.

```

from EnglishBot import bind_context

@bind_context
def start (update, context):
    # Now we have access to everything from one place
    # For example, we can easily add a new user to the database
    # Check if the user is in our database
    if not context._bot.user_exist (context.t_id):
        # For example, add some important notifications
        context.set ('push_notification', True)
        # And then add the user to the database
        context._bot.new_user (context.t_id, context.t_ln)

    return update.message.reply_text ('Welcome')
# ...

```

I decoratori sono il nostro tutto

Molto spesso, si scopre che il nome della funzione coincide con il nome del comando a cui vogliamo aggiungere un gestore. Perché non utilizzare questa funzione statistica per i tuoi scopi egoistici.

```

class EnglishBot:
    # ...

    def add_command_handler (self, func: function, name = None) -> None:
        """
        Function that adds a command handler
        """

        name = name or func.__name__
        self.dp.add_handler (CommandHandler (name, func))

# ...

# In the main file:
tbot.add_command_handler (start) # Instead of tbot.add_command_handler (start, 'start')

```

Sembra bello, ma non funziona. Riguarda il decoratore `bind_context`, che restituirà sempre il nome della funzione wrapper. Correggi questo malinteso.

```

import functools

def bind_context (func):
    # functools.wraps from stdlib saves signatures since Python 3.4
    @functools.wraps (func)
    def wrapper (update, context):
        context._bot.bind_user_data (update, context)
        return func (update, context)
    return wrapper

```

Ci sono molti gestori di messaggi nel bot che, in base alla progettazione, dovrebbero annullare il comando quando si immette zero. Inoltre devo eliminare tutti i post modificati.


```

import functools

END = -1

def zero_exiter (func):
    @ functools.wraps (func)
    def wrapper (update, context):
        if update.to_dict () ['message']. get ('text', None) == '0':
            update.message.reply_text ('Sending some message')
            return END

        return func (update, context)
    return wrapper

def skip_edited (func):
    @ functools.wraps (func)
    def wrapper (update, context):
        # This works in all cases, because None returned
        # in the conversation_handler stack, leaves the function in its current state
        if not update.to_dict (). get ('edited_message', None):
            return func (update, context)
    return wrapper

```

Non dimentichiamo allo stesso tempo del decoratore più importante - @run_async su cui si basa l'asincronia. Ora raccogliamo la funzione pesante.

```

from telegram.ext.dispatcher import run_async
from EnglishBot import skip_edited

@run_async
@skip_edited
def heavy_function (update, context):
    # ...
    # A heavy computing function that needs asynchrony
    # Has post edit protection
    # ...

```

Ricorda che l'asincronia è una spada Jedi, ma con questa spada puoi farlo rapidamente [uccisione te stesso](#).

A volte i programmi si bloccano senza inviare nulla al registro. @logger.catch, l'ultimo decoratore del nostro elenco, garantisce che qualsiasi errore venga segnalato correttamente a un logger.

```

from loguru import logger

@logger.catch
def heavy_function2 (update, context):
    # ...
    # Another heavy computing function on which a program may hang
    # ...

```

Pannello di Amministrazione

Implementiamo un pannello di amministrazione con la possibilità di ricevere/eliminare i log e inviare un messaggio a tutti gli utenti.


```

from EnglishBot import bind_context
from Cache import file_is_empty
from telegram import ReplyKeyboardMarkup
from loguru import logger

LOG_FILE = 'logs.log'
SENDING, MAIN, END = range(1, -2, -1)

buttons = ReplyKeyboardMarkup (
    [('Get logs', 'logs'), ('Clear logs', 'clear')],
    [('Send message', 'send')],
    # [...],
)

@bind_context
def admin_panel (update, context):
    # Check user admin rights
    if not context._bot.aces_check (context.t_id):
        # A good security practice is to show the user that such a command does not exist
        return update.message.reply_text (f'Unknown command {update.message.text} ')

    update.message.reply_text ('Choose an option:', reply_markup = buttons)

return MAIN

```

logs methods

```
def get_logs (update context).
    if file is empU. (LOG FILE).
        update.cd1back quo.'.rep1>' text ("Logs are empU.") else.
        = Show document loading
        context.bot.send chat acti <xi (chat id = context.t icL action = 'upload docum.cut')
        context.bot.sendDocument (chat i d = context.t icL docimi cut = open (L OG FILE
        'rb') name = L Ohr FILE
timeexit = 1000)
```

= Since we do not close the admin panel you need to remove the download icon on the button update.cd1back quell'.answer (text = "
= Basicly this is unnecessary. As I said earlier None does not change the position of the handler
= But this was the code looks much more readable return ñIAIN

```
def logs_d ear (update context).
    with open (LOG FILE 'u") as file. update.cd1back quo.'.rep1>' text ('O ear ed)
        update.cd1back quo.'.answer (text = ")

    return ñIAIN
```

= Send methods

```
def take_message (update context).
    update.cd1back quell'.rep1>' text ('Send message) update.cd1back quell'.answer (text =
    ")

    return SENDING
```

```
!fi ero exiter
def send_d1 (updata context).
    count = 0
```

```

Get user identifiers from the database for id in list(context.bot.get_user_ids()).
= It may happen that some user has added a bet to the emergency
= Then when trying to send him a message an error will be caught to:
= if we do not send a message to ourselves if id == context.bot.get_me().id: continue
= Be sure to use markdown to save message formatting
context.bot.send_message(context.chat_id, text=update.message.text, markdown
parse_mode = 'markdown') count += 1
except: pass

update.callback_query.text = f'Sent to {count} people'
update.callback_query.answer(text='')

```

```

# ...

# In the main file:
tbot.add_conversation_handler(
    entry_points=[('admin', admin)],
    # The regularity for send_alk allows you to process any message that does not start with /
    states=[[(logs, '^ logs $'), (logs_clear, '^ clear $'), (send, '^ send $'), [(send_alk, '@ ^((?!.*((^|/)+)).*)(.+)$')]]]
)

```

La funzione `add_conversation_handler` ti consente di aggiungere un gestore di conversazione in modo minimalista:

```

class EnglishBot:
    # ...

    def add_conversation_handler(self, entry_points: list, states: list, fallbacks: list) -> None:
        fallbacks = [CommandHandler(name, func) for name, func in fallbacks]
        entry_points = [CommandHandler(name, func) for name, func in entry_points]
        r_states = {}

        for i in range(len(states)):
            r_states[i] = []

            # Each array describes functions of one state
            for func, pattern in states[i]:
                # If the regular starts with the @ symbol, then we add a message handler
                # Otherwise, a regular button handler
                if pattern[0] == '@':
                    r_states[i].append(MessageHandler(Filters.regex(pattern[1:]), func))
                else:
                    r_states[i].append(CallbackQueryHandler(func, pattern=pattern))

        conv_handler = ConversationHandler(entry_points=entry_points, states=r_states, fallbacks=fallbacks)
        dp.add_handler(conv_handler)

```

Funzionalità principale

Insegniamo al nostro bot ad aggiungere nuove parole ai dizionari degli utenti.

```

from EnglishBot import bind_context, skip_edited, zero_exiter
from youtube_transcript_api import YouTubeTranscriptApi
from telegram.ext.dispatcher import run_async

START_OVER, ADDING, END = range(1, -2, -1)

re_youtube = re.compile ('^ (http (s)? : \ / \ /)? ((w) {3}.)?youtu (be | .be)? (\ . com)? \ / .+ ' )
re_text = re.compile ('^ [az] {3,20} $')

def is_youtube_link (link: str) -> bool:
    if re_youtube.match (link) is not None: return True

def clear_text (text: str) -> str:
    bad_symbols = '! @ # % $ ^ & * () _ + 1234567890 - = / | \ \? > < , . " : ' ~ [] {} '

```

for s in bad_symbols:

```

        text = text.replace(s, "")

    return text.strip()

!@skip edited
:@bind_context
def add_words(update, context):
    update.message.reply_text('Enter text:')
    return ADDING

:Handle asynchronous
:@skip_edited
:@zero_exiter
def parse_text(update, context):
    DDMMDDZéng takes some time, so if you need to tell the user that the 'thing is fine
    message = update.message.reply_text('Loading...')

    if is_outube_link(update.message.text):
        = If the video is invalid or there are no subtitles in it, then we will catch an error
        to:
            transcript_list = YouTubeTranscriptApi.list_transcripts(get_video_id(update.message.text))
            t = transcript_list.find_transcript('en')
            txt = clear_txt(''.join([i[0] for i in t[0]]))
        except:
            message.reply_text('Could not find subtitles. To be again:')
            return ADDING
        else:
            _text = clear_txt(update.message.text).split()

    = We get a link to the words the user already has
    T&ds.' CDotext. bDget ckt 'ords (cDttWttid)
    1 'ards to be added
    good_words = []
    = Discarded words
    bad_words = []

    = First, we discard duplicates
    for word in set(_text):
        = Then we check the correctness of the word regular
        z = re_text.match(word)
        if z:
            = Add a word only if it is not already in the user's list
            if z.group() not in _words:
                good_words.append(word)
        else:
            bad_words.append(word)

    = The Dnl ' thi Hg left 1 S tD suggest the user to Add ends
    = And then add them to the dictionary

.. In the main file
    bot.add_conversation_handler(
        entry_points=[('add_words', add_words)],
    states = [
        [(parse_text, '@ ^ ((?! * ((^ \ /) +)). *) (. + $)'],
        # ...
    ])

```

Imballiamo il bot

Prima di impacchettare il nostro bot, aggiungi il supporto proxy e modifica i livelli di registro dalla console.

In the main file:

```
import argparse
```

```
parser = argparse.ArgumentParser()
```

```
parser.add_argument('-l', '--level', default = 'INFO',
```

```
    choices = ['TRACE', 'DEBUG', 'INFO', 'SUCCESS', 'WARNING', 'ERROR', 'CRITICAL'],
```

```
    help = 'Allows you to enable the bot in a given mode')
```

```
parser.add_argument('-p', '--proxy', help = 'Allows you to enable bot with proxy')
```

```
args = parser.parse_args()
```

```
config = {
```

```
    'handlers': [
```

```
        {'sink': sys.stdout, 'level': args.level},
```

```
        # It's not a mistake. In the file I collect logs of level DEBUG and higher
```

```
        {'sink': 'logs.log', 'serialize': False, 'level': 'DEBUG'},
```

```
    ],
```

```
}
```

```
logger.configure(** config)
```

```
if args.proxy:
```

```
    t_proxy = {'proxy_url': args.proxy, 'read_timeout': 1000, 'connect_timeout': 1000}
```

```
    # ...
```

```
    # Proxies for other services
```

```
    # ...
```

```
else:
```

```
    t_proxy = None
```

Python 3.5+ supporta il [capacità di comprimere](#) una directory in un singolo file eseguibile. Cogliamo l'occasione per poter distribuire facilmente il tuo ambiente di lavoro su qualsiasi VPS. Per prima cosa, ottieni il file delle dipendenze. Se stai usando un ambiente virtuale, puoi farlo con un comando: `pip freeze > require.txt`. Se il progetto non ha un ambiente virtuale, dovrai armeggiare un po'. Puoi provare a usare `pip freeze` e isolare manualmente tutti i pacchetti necessari. Tuttavia, se sul sistema sono installati troppi pacchetti, questo metodo non funzionerà. La seconda opzione è quella di utilizzare soluzioni già pronte, [ad esempio, pipreq](#).

Ora che il nostro file delle dipendenze è pronto, possiamo comprimere la nostra directory in a . pyzfile. Per fare ciò, inserisci il comando `py -m zipapp "ПУТЬ_К_КАТАЛОГУ" -m "ИМЯ_ГЛАВНОГО_ФАЙЛА:ГЛАВНАЯ_ФУНКЦИЯ" -o bot.pyz`, creerà il file `bot.pyz` nella cartella del progetto. Si prega di notare che il codice in `init . py` deve essere racchiuso in qualche funzione, altrimenti il file eseguibile sarà impossibile da compilare.


```
# Sample __init__.py file
# py -m zipapp "PATH_TO_CATALOG" -m "__init__.py:main" -o bot.pyz

def main():
    # ...

if __name__ == '__main__':
    main()
```

Avvolgiamo i file nell'archivio zip bot.zip require.txt bot.pyz e lo inviamo al nostro VPS.

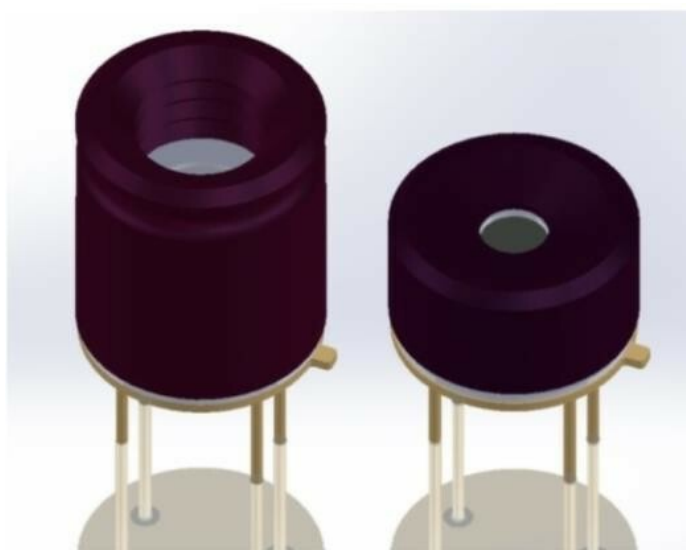
5. LA TERMOCAMERA SUL PI DI LAMPONE

W sul famoso sito cinese sono comparsi i famosi moduli di imaging termico. È possibile assemblare una cosa esotica e, possibilmente, anche utile?

- una termocamera fatta in casa? Perché no, come se Raspberry stesse mentendo? da qualche parte. Cosa ne è venuto fuori - te lo dirò sotto il taglio.

MLX90640. Che cos'è?

E questa, infatti, è una matrice per immagini termiche con a bordo un microcontrollore. Produzione della società precedentemente sconosciuta Melexis. La matrice di imaging termico ha una dimensione di 32 per 24 pixel. Non è molto, ma interpolando l'immagine, sembra essere sufficiente per distinguere almeno qualcosa.



Sono disponibili due tipi di sensori, i cui casi differiscono per l'angolo di visione della matrice. Una struttura più tozza A si affaccia sul mondo esterno con un angolo di 110 (orizzontale) a 75 (verticale) gradi. B - sotto i 55 di 37,5 gradi, rispettivamente. La custodia del dispositivo ha solo quattro uscite - due per l'alimentazione, due per la comunicazione con il dispositivo di controllo tramite l'interfaccia I2C. Le schede tecniche interessate possono essere [scaricate qui](#).

E poi cos'è il GY-MCU90640?

I compagni cinesi hanno messo a bordo l'MLX90640 con un altro microcontrollore a bordo (STM32F103). Apparentemente, per una più facile gestione della matrice. L'intera fattoria si chiama GY-MCU90640. E costa al momento dell'acquisizione (fine dicembre 2018) nella regione di 5 migliaia di \$. Come segue:



Come vedi, ci sono due tipi di schede, con una versione stretta o grandangolare del sensore a bordo.

Qual è la versione migliore per te? Una buona domanda, purtroppo, l'ho fatta solo dopo che i moduli erano già stati ordinati e ricevuti. Per qualche ragione, in questo momento degli ordini, non ho prestato attenzione a queste sfumature. Ma invano.

Una versione più ampia sarà utile su robot semoventi o in sistemi di sicurezza (il campo visivo sarà più ampio). Secondo le schede tecniche, ha anche meno rumore e una maggiore precisione di misurazione.

Ma per le attività di visualizzazione, consiglieri di più una versione più "a lungo raggio" di B. Per una ragione molto significativa. In futuro, durante le riprese, può essere distribuito (manualmente o su una piattaforma con un'unità) e scattare "foto" composite, aumentando così la più che modesta risoluzione di 32 per 24 pixel. Raccoglie immagini termiche 64 per 96 pixel, ad esempio. Va bene, in futuro le foto saranno della versione grandangolare A.

Connetti a Raspberry PI

Esistono due modi per controllare il modulo di imaging termico:

1. Accorciare il ponticello "SET" sulla scheda e utilizzare I2C per contattare direttamente il microcontrollore interno MLX90640.
2. Lascia stare il jumper e comunica con il modulo

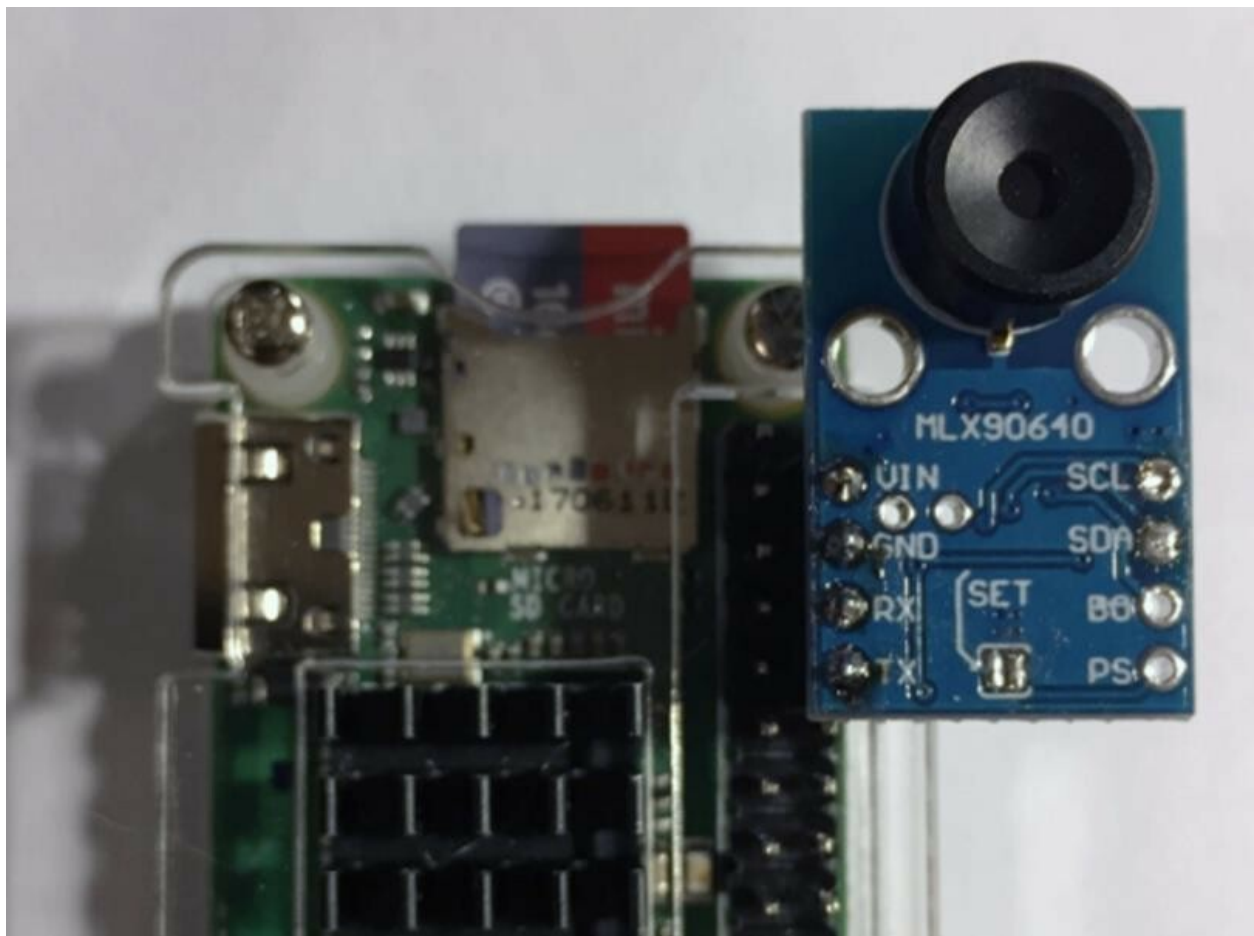
tramite un'interfaccia simile installata sulla scheda STM32F103 via RS-232.

Se scrivi in C++, probabilmente sarà più conveniente ignorare il microcontrollore aggiuntivo, cortocircuitare il ponticello e utilizzare l'API del produttore, che si trova qui.

Anche i pitoni umili possono scegliere la prima strada. Sembra che ci siano un paio di librerie Python ([qui](#) e [qui](#)). Ma sfortunatamente, non uno ha funzionato per me.

I Python avanzati possono scrivere un driver di controllo del modulo in Python. La procedura per ottenere un telaio è descritta in dettaglio nella scheda tecnica. Ma poi dovrai prescrivere tutte le procedure di calibrazione, che sembra leggermente gravosa. Pertanto, ho dovuto andare nella seconda strada. Si è rivelato moderatamente spinoso, ma abbastanza passabile.

Grazie all'intuizione degli ingegneri cinesi o solo a una felice coincidenza, lo scialle si è rivelato avere una posizione perfetta delle conclusioni:



Resta solo da mettere il blocco e inserire la sciarpa nel connettore lampone. Sulla scheda è installato un convertitore da 5 a 3 Volt, quindi sembra che nulla minacci i delicati terminali Rx e Tx di Raspberry.

Va aggiunto che anche la connessione secondo la prima opzione è possibile, ma richiede più manodopera e abilità di saldatura. La scheda deve essere installata sull'altro lato del connettore Raspberry (mostrato nella foto del titolo di questo post).

Morbido

Su un noto sito cinese, viene offerto un tale miracolo per accedere al GY-MCU90640:

Molto probabilmente, dovrebbe esserci una descrizione del protocollo di interazione con il microcontrollore installato sulla scheda, in base al quale funziona questo prodotto software! Dopo una breve conversazione con il venditore di sciarpe (rispetto a questi rispettati signori), mi è stato inviato un tale protocollo. È apparso in pdf e cinese puro.

Grazie al traduttore di Google e al copia-incolla attivo, dopo circa un'ora e mezza il protocollo è stato decifrato, chiunque può leggerlo su [Github](#). Si è scoperto che la sciarpa comprende sei comandi di base, tra i quali c'è una richiesta di frame sulla porta COM.

Ogni pixel della matrice è, infatti, il valore della temperatura dell'oggetto che questo pixel sta guardando. La temperatura in gradi Celsius per 100 (numero a doppio byte). C'è anche una modalità speciale in cui la sciarpa invierà frame dalla matrice al Raspberry 4 volte al secondo.

Copyright (c) 2019

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

IL SOFTWARE È FORNITO 'COME È', PRIVO DI GARANZIA DI QUALUNQUE TIPO, ESPRIMERE O IMPLICITA. COMPRESO MA NON LIMITATA ALLE GARANZIE DI COMMERCIALIZZABILITÀ, FITNESS PER UN PARTICOLARE SCOPO E NON VIOLAZIONE. IN NO EVENTO DEVE IL AUTORE O DIRITTO D'AUTORE TITOLARI ESSERE RESPONSABILE PER QUALUNQUE

RECLAMO, DANNI O DI ALTRE RESPONSABILITÀ, SE IN ANNAZIONE DEL CONTRATTO, ILLECITO O
ALTRIMENTI, SORGENTE A PARTIRE DAL, FUORI DI O IN CONNESSIONE CON
SOFTWARE O L'USO O ALTRI TRATTAMENTI NELLA SOFTWARE. " "

```
import numpy as np
import cv2
```

D funzione a ottenere Emissività a partire dal MCU

```
def get_emissivity():
    ser.write((semi D).to_bytes([0xA5, D*55, 0x01, 0xFB]))
```

leggere = ser.read(4) Restituzione leggere [2] / 100

D funzione a ga temperature a partire dal MCU (Centigrado livello X 100)

Rif ga_temp_oray (D):

ottenere ambiente domare

T_a = (iot (d[1540]) + int (d [1541]) • 256) / 100

ottenere crudo Vettore di pixel temperie

raw_data = D [4: 1540]

T_array = np.frombuffer(raw_data, dtype=np.int16)

return T_a, T_array

funzione a convertire temperie a pixel o a Immagine

def temp_to_image(f):

norma = np.int8 ((f, 100 - Gemellare) • 255, (Tmax-Tmin))

norm.shy = (24.32)

```
##### Main cycle #####
```

```
# Color map range
```

T_n = 40

Tmin = 20

```
print('Configuring Serial port')
ser = serial.Serial('dev:/dev/ttyS0')
ser.baudrate = 115200
```

```
=set frequency of module to 4 Hz
ser.write(b'\x05\x25\x01\xCB')
time.sleep(0.1)
```

```
=Starting automatic data collection
ser.write(b'\x05\x35\x02\xD')
t0 = time.time()
```

```
while True:
```

```
=waiting for data frame
    data = ser.read(1544)
```

```
=The data is ready! let's handle it!
    To temp array = get temp array (data)
    temp_img = temp_to_image(temp_array)
```

```
=Image processing
    img = cv2.cvtColor(temp_img, cv2.COLOR_BGR2RGB)
    img = cv2.resize(img, (320, 240))
    img = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
```

```

text=Tmin={,+.lf) Tmax={,+.lf) FPS={,.2f}'.format(temp
array'.min() ."100 temp array'.max() ."100
1. "(time.time()-t0)
cv2.putText(img,text,(5,15),cv2.FONT_HERSHEY_SIMPLEX,0.45,(0
0,0,1))
cv2.imshow('Output',img)

```

```

* if 's' is pressed - saving of picture key=cv2.waitKey(1) & 0xFF
if key==ord('s'):
    fname='pic'+dt.datetime.now().strftime("%Y-%m-%d-%H-%M-%S")+'.jpg'
    cv2.imwrite(fname,img)
    print('Saving image',fname)
    t0=time.time()

```

```
except KeyboardInterrupt:
```

```
* to terminate the cycle
```

```
ser.write(serial.b'\x05\x35\x01\xDB')) ser.close()
```

```
cv2.destroyAllWindows() print('Stopped')
```

```
= just in case ser.close()
```

```
cv2.destroyAllWindows()
```

Risultati

Lo script esegue il polling della matrice di imaging termico e invia i frame alla console del monitor su cui è collegato il Raspberry PI, quattro volte al secondo. Questo è sufficiente per non provare un disagio significativo quando si riprendono gli oggetti. Per visualizzare il frame, viene utilizzato il pacchetto OpenCV. Alla pressione del tasto "s" le "mappe termiche" della termografia in formato jpg vengono salvate nella cartella con lo script.

Per maggiori informazioni, ho dedotto le temperature minime e massime sul telaio. Cioè, guardando il colore, puoi vedere qual è approssimativamente la temperatura degli oggetti più riscaldati o refrigerati. L'errore di misurazione è di circa un grado con un lato più grande. Il range termico è impostato da 20 a 40 gradi. Uscire dallo script premendo Ctrl + C.

Lo script funziona approssimativamente allo stesso modo sia su Raspberry Pi Zero W che su Pi 3 B+. Ho installato il server VNC sullo smartphone. Pertanto, raccogliendo lamponi collegati a un power bank e uno smartphone con VNC in esecuzione, è possibile ottenere una termocamera portatile con la possibilità di salvare le immagini termiche. Forse questo non è del tutto conveniente, ma abbastanza funzionale.

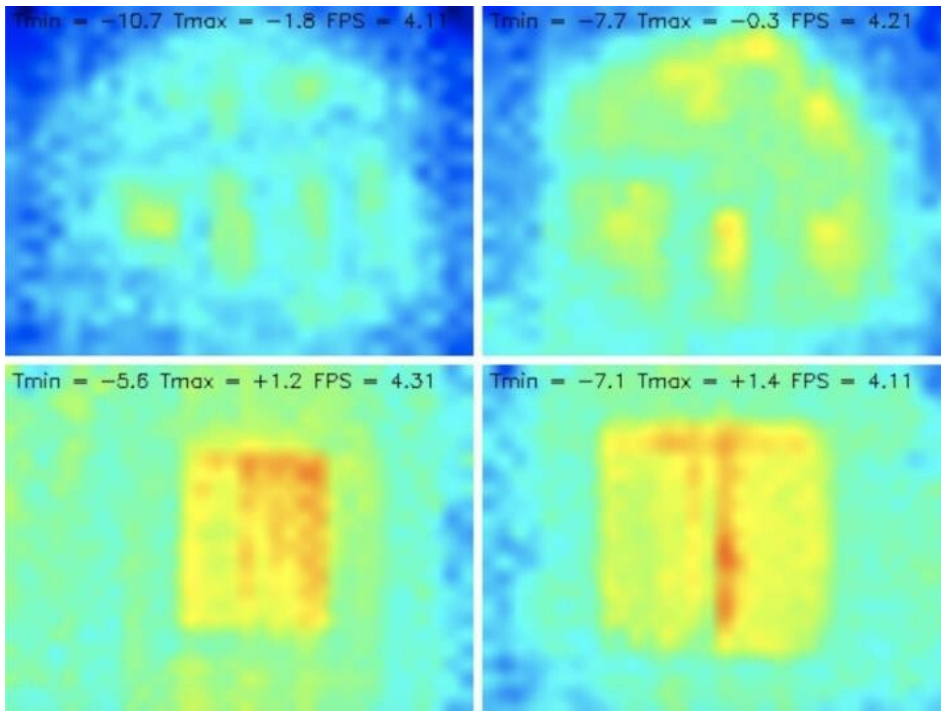
Dopo il primo avvio è possibile una misurazione errata della temperatura massima. In questo caso, è necessario uscire dallo script e rieseguirlo.

Questo è tutto per oggi. L'esperimento con una termocamera fatta in casa

si è rivelato vincente. Con l'aiuto di questo dispositivo, ad esempio, è possibile condurre da soli un'ispezione termica della casa.

A causa del contrasto di temperatura inferiore rispetto all'interno, le immagini non erano molto istruttive. Nella foto sopra, tutta la casa è su due lati. In basso - foto di diverse finestre.

Nel codice, ho modificato solo l'intervallo di temperatura. Invece di +20 ... + 40, ho impostato -10 ... + 5.



6. TROVARE UN PARCHEGGIO GRATUITO CON PITONE



io vivere in una vera città. Ma, come in tanti altri, la ricerca di un parcheggio si trasforma sempre in una prova. Gli spazi liberi occupano rapidamente, e anche se hai il tuo, sarà difficile per gli amici chiamarti perché non avranno un posto dove parcheggiare.

Quindi ho deciso di puntare la fotocamera fuori dalla finestra e utilizzare il deep learning in modo che il mio computer mi dica quando lo spazio è disponibile:

Può sembrare complicato, ma scrivere un prototipo funzionante con il deep learning è facile e veloce. Tutti i componenti necessari sono già presenti: devi solo sapere dove trovarli e come assemblarli.

Quindi divertiamoci e scriviamo un accurato sistema di notifica del parcheggio gratuito usando Python e il deep learning

Scomporre il compito

Quando abbiamo un compito difficile che vogliamo risolvere con l'aiuto dell'apprendimento automatico, il primo passo è suddividerlo in una sequenza di compiti semplici. Quindi possiamo utilizzare vari strumenti per risolverli. Combinando diverse soluzioni semplici, otteniamo un sistema capace di qualcosa di complesso.

Ecco come ho interrotto il mio compito:

Il flusso video dalla webcam diretto alla finestra entra nell'input del nastro trasportatore: attraverso la pipeline, trasmetteremo ogni fotogramma del video, uno alla volta.

Il primo punto è riconoscere tutti i possibili posti auto nel telaio. Prima di poter cercare i luoghi non occupati, dobbiamo capire in quali parti dell'immagine c'è parcheggio.

Quindi su ogni frame devi trovare tutte le auto. Questo ci permetterà di tracciare il movimento di ogni macchina da un telaio all'altro.

Il terzo passo è determinare quali posti sono occupati dalle macchine e

che non lo sono. Per fare ciò, combinare i risultati dei primi due passaggi.

Infine, il programma dovrebbe inviare un avviso quando il parcheggio diventa libero. Questo sarà determinato dai cambiamenti nella posizione delle macchine tra i fotogrammi del video.

Ciascuno dei passaggi può essere completato in modi diversi utilizzando tecnologie diverse. Non esiste un unico modo giusto o sbagliato per comporre questo trasportatore: diversi approcci avranno i loro vantaggi e svantaggi. Affrontiamo ogni passaggio in modo più dettagliato.

Riconosciamo i parcheggi

Ecco cosa vede la nostra fotocamera:



Dobbiamo scansionare questa immagine in qualche modo e ottenere un elenco di posti per parcheggiare:



La soluzione "in fronte" sarebbe semplicemente codificare manualmente le posizioni di tutti i parcheggi invece di riconoscerli automaticamente. Ma in questo caso, se spostiamo la telecamera o vogliamo cercare parcheggi in un'altra strada, dovremo rifare tutta la procedura. Sembra così così, quindi cerchiamo un modo automatico per riconoscere i parcheggi.

In alternativa, puoi cercare i parchimetri nell'immagine e presumere che ci sia un parcheggio accanto a ciascuno di essi:



Tuttavia, con questo approccio, non tutto è così liscio. In primo luogo, non

ogni posto auto ha un parchimetro, e in effetti siamo più interessati a trovare parcheggi per i quali non devi pagare. In secondo luogo, la posizione del parchimetro non ci dice nulla su dove si trova il parcheggio, ma ci permette solo di fare un'ipotesi.

Un'altra idea è quella di creare un modello di riconoscimento degli oggetti che cerchi segni di parcheggio disegnati sulla strada:

Ma questo approccio è così così. Innanzitutto, nella mia città, tutti questi marchi sono molto piccoli e difficili da vedere a distanza, quindi sarà difficile rilevarli utilizzando un computer. In secondo luogo, la strada è piena di ogni sorta di altre linee e segni. Sarà difficile separare i segnali di parcheggio dai divisori di corsia e dagli attraversamenti pedonali.

Quando incontri un problema che a prima vista sembra complicato, prenditi qualche minuto per trovare un altro approccio alla risoluzione del problema, che ti aiuterà ad aggirare alcuni problemi tecnici. Qual è il parcheggio? Questo è solo un posto dove un'auto è parcheggiata a lungo. Forse non abbiamo affatto bisogno di riconoscere i parcheggi. Perché non riconosciamo solo le auto ferme da molto tempo e non presumiamo che siano ferme in un parcheggio?

In altre parole, i parcheggi si trovano dove le auto sostano a lungo:

Quindi, se siamo in grado di riconoscere le auto e scoprire quali di esse non si muovono tra i fotogrammi, possiamo indovinare dove si trovano i parcheggi. Semplicemente così: passiamo al riconoscimento delle auto!

Riconoscere le auto

Riconoscere le auto su un fotogramma video è un classico compito di riconoscimento degli oggetti. Esistono molti approcci di apprendimento automatico che potremmo utilizzare per il riconoscimento. Eccone alcuni in ordine dalla "vecchia scuola" alla "nuova scuola":

- Puoi addestrare il rilevatore in base a HOG (Istogramma dei gradienti orientati, istogrammi dei gradienti direzionali) e percorrerlo attraverso l'intera immagine per trovare tutte le auto. Questo vecchio approccio, che non utilizza l'apprendimento profondo, funziona in modo relativamente rapido ma non si adatta molto bene a macchine situate in modi diversi.

- Puoi addestrare un rilevatore basato sulla CNN (Convolutional Neural Network, una rete neurale convoluzionale) e percorrere l'intera immagine fino a trovare tutte le auto. Questo approccio funziona in modo preciso, ma non così efficiente poiché dobbiamo scansionare l'immagine più volte utilizzando la CNN per trovare tutte le macchine. E sebbene possiamo trovare macchine posizionate in modi diversi, abbiamo bisogno di molti più dati di addestramento rispetto a un rilevatore HOG.
- Puoi utilizzare un nuovo approccio con il deep learning come Mask R-CNN, Faster R-CNN o YOLO, che combina l'accuratezza della CNN e una serie di trucchi tecnici che aumentano significativamente la velocità di riconoscimento. Tali modelli funzioneranno in tempi relativamente brevi (sulla GPU) se disponiamo di molti dati per l'addestramento del modello.

Nel caso generale, abbiamo bisogno della soluzione più semplice, che funzionerà come dovrebbe e richiederà la minor quantità di dati di addestramento. Non è necessario che questo sia l'algoritmo più recente e più veloce. Tuttavia, in particolare nel nostro caso, Mask R-CNN è una scelta ragionevole, anche se unica e veloce.

L'architettura Mask RCNN è progettata in modo tale da riconoscere gli oggetti nell'intera immagine, spendendo efficacemente risorse e non utilizza l'approccio della finestra scorrevole. In altre parole, funziona abbastanza velocemente. Con una moderna GPU, possiamo riconoscere gli oggetti nel video in alta risoluzione a una velocità di diversi fotogrammi al secondo. Per il nostro progetto, questo dovrebbe essere sufficiente.

Inoltre, Mask R-CNN fornisce molte informazioni su ciascun oggetto riconosciuto. La maggior parte degli algoritmi di riconoscimento restituisce solo un riquadro di delimitazione per ogni oggetto. Tuttavia, Mask R-CNN non solo ci darà la posizione di ciascun oggetto, ma anche il suo contorno (maschera):

Per addestrare Mask R-CNN, abbiamo bisogno di molte immagini degli oggetti che vogliamo riconoscere. Potremmo uscire, fotografare le auto e segnarle nelle fotografie, il che richiederebbe diversi giorni di lavoro. Fortunatamente, le auto sono uno di quegli oggetti che le persone spesso vogliono riconoscere, quindi esistono già diversi set di dati pubblici con immagini di auto.

Uno di questi è il popolare SOCO [set di dati](#) (abbreviazione di Common Objects In Context), che ha immagini annotate con maschere di oggetti. Questo set di dati contiene oltre 12.000 immagini con macchine già etichettate. Ecco un'immagine di esempio dal set di dati:

Tali dati sono eccellenti per addestrare un modello basato su Mask R-CNN.

Ma tenete i cavalli, ci sono notizie ancora migliori! Non siamo i primi a voler addestrare il proprio modello utilizzando il set di dati COCO: molte persone lo avevano già fatto prima di noi e hanno condiviso i loro risultati. Pertanto, invece di addestrare il nostro modello, possiamo prenderne uno già pronto che possa già riconoscere le auto. Per il nostro progetto, utilizzeremo [il modello open source di Matterport.](#)

Se diamo l'immagine dalla fotocamera all'input di questo modello, questo è ciò che otteniamo già "fuori dalla scatola":

Il modello ha riconosciuto non solo automobili ma anche oggetti come semafori e persone. È strano che abbia riconosciuto l'albero come una pianta d'appartamento.

Per ogni oggetto riconosciuto, il modello Mask R-CNN restituisce quattro cose:

- Tipo di oggetto rilevato (intero). Il modello COCO pre-addestrato è in grado di riconoscere 80 diversi oggetti di uso quotidiano come automobili e camion. Un elenco completo è disponibile [qui.](#)
- Il grado di fiducia nei risultati del riconoscimento. Più alto è il numero, più forte è la fiducia del modello nel corretto riconoscimento dell'oggetto.
- Casella ricca per un oggetto sotto forma di coordinate XY dei pixel nell'immagine.
- Una "maschera" che mostra quali pixel all'interno del riquadro di delimitazione fanno parte dell'oggetto. Usando i dati della maschera, puoi trovare il contorno dell'oggetto.

Di seguito è riportato il codice Python per rilevare il riquadro di delimitazione per le macchine che utilizzano i modelli Mask R-CNN e OpenCV pre-addestrati:

```
import numpy as np

import mrcnn.config import mrcnn.utils
from mrcnn.model import MaskRCNN from pathlib import Path

# The configuration for the MaskRCNN libm. u411 use class
MaskRCNNConfig(mrcnn.config.Config):
    DEVICE = "coco_retrained_model_config" EvfAGES PER GPL = 1
    GPL COLNT = 1
    CLASSES = 1 + 80 = in the COCO dataset there are 80 classes +
    1 background class. DETECTION UHN CONFIDENCE = 0.6
```


=filter the list of recognition results so that only cars remain. def get_car_boxes(boxes class_ids):

car_boxes=[]

for i, box in enumerate(boxes):

=If the found object is not a car then skip it. if class_ids[i] in [3, 8, 6]:

car_boxes.append(box) return np.array(car_boxes)

=The root directory of the project. ROOT_DIR = Path(".")

=Directory for saving logs and trained model. MODEL_DIR = ROOT_DIR / "logs"

* Load path to the file with trained weights.

COCO_MODEL_PATH = ROOT_DIR / "mask_rcnn_coco.h5"

=Download COCO dataset if necessary. if not COCO_MODEL_PATH.exists():

utils.download_trained_weights(COCO_MODEL_PATH)

=Directory with images for processing. EVAL_DIR = ROOT_DIR / "images"

=video file or camera for processing - insert a value of 0 if you want to use a camera not a video file. VIDEO_SOURCE = "test_images/parking.mp4"

=Create a flask-RCNN model in output mode.

model = flask_Wrapper(mode="inference" model_dir=MODEL_DIR config=
flask_RCUNetConfig)

=Download the pre-trained model.

model.load_weights(COCO_MODEL_PATH, name=True)

* Location of parking spaces.

parked_cars = None

```

# Download the video file for which we want to run recognition.
video_capture = cv2.VideoCapture (VIDEO_SOURCE)

# We loop through each frame.
while video_capture.isOpened():
    success, frame = video_capture.read()
    if not success:
        break

    # Convert the image from the BGR color model (used by OpenCV) to RGB.
    rgb_image = frame[:, :, :: - 1]

    # We supply the image of the Mask R-CNN model to get the result.
    results = model.detect ([rgb_image], verbose = 0)

    # Mask R-CNN assumes that we recognize objects in multiple images.
    # We transmitted only one image, so we extract only the first result.
    r = results [0]

    # The variable r now contains recognition results:
    # - r ['rois'] - bounding box for each recognized object:
    # - r ['class_ids'] - identifier (type) of the object:
    # - r ['scores'] - degree of confidence:
    # - r ['masks'] - masks of objects (which gives you their outline).

    # Filter the result to get the scope of the car.
    car_boxes = get_car_boxes (r ['rois'], r ['class_ids'])

    print (" Cars found in frame of video:")

    # Display each frame on the frame.
    for box in car_boxes:
        print (" Car:", box)

        y1, x1, y2, x2 = box

        # Draw a frame.
        cv2.rectangle (frame, (x1, y1), (x2, y2), (0, 255, 0), 1)

    # Show the frame on the screen.
    cv2.imshow ('Video', frame)

    # Press 'q' to exit.
    if cv2.waitKey (1) & 0xFF == ord ('q'):
        break

# We clear everything after completion.
video_capture.release ()
cv2.destroyAllWindows ()

```

Dopo aver eseguito questo script, sullo schermo apparirà un'immagine con una cornice attorno a ciascuna macchina rilevata: Inoltre, nella console verranno visualizzate le coordinate di ciascuna macchina:

```

Cars found in frame of video:
Car: [492 871 551 961]
Car: [450 819 509 913]
Car: [411 774 470 856]

```

Così abbiamo imparato a riconoscere le auto nell'immagine.

Riconosciamo i parcheggi vuoti

Conosciamo le coordinate dei pixel di ogni macchina. Guardando attraverso diversi fotogrammi consecutivi, possiamo determinare rapidamente quale delle auto non si è mossa

e supponiamo che ci siano parcheggi. Ma come capire che l'auto è uscita dal parcheggio?

Il problema è che i telai delle macchine si sovrappongono parzialmente tra loro:

Pertanto, se si immagina che ogni telaio rappresenti un posto auto, può risultare che sia parzialmente occupato dalla macchina, quando in realtà è vuota. Dobbiamo trovare un modo per misurare il grado di intersezione di due oggetti per cercare solo i frame "più vuoti".

Useremo una misura chiamata Intersection Over Union (rapporto tra area di intersezione e area totale) o IoU. IoU può essere trovato calcolando il numero di pixel in cui due oggetti si intersecano e dividono per il numero di pixel occupati da questi oggetti:

Quindi possiamo capire come la cornice di delimitazione dell'auto si intersechi con la cornice del parcheggio. rendono facile determinare se il parcheggio è gratuito. Se l'IoU è basso, come 0,15, l'auto occupa una piccola parte dello spazio di parcheggio. E se è alto, come 0,6, significa che l'auto occupa la maggior parte dello spazio e non puoi parcheggiare lì.

Poiché IoU viene utilizzato abbastanza spesso nella visione artificiale, è molto probabile che le librerie corrispondenti implementino questa misura. Nella nostra libreria Mask R-CNN, è implementata come funzione `mrcnn.utils.compute_overlaps()`.

Se disponiamo di un elenco di riquadri di delimitazione per posti auto, puoi aggiungere un controllo per la presenza di auto in questo quadro aggiungendo una o due righe intere di codice:

```
# Filter the result to get the scope of the car.
car_boxes = get_car_boxes(r['rois'], r['class_ids'])

# We look how much cars intersect with well-known parking spaces.
overlaps = mrcnn.utils.compute_overlaps(car_boxes, parking_areas)

print(overlaps)
```

Il risultato dovrebbe essere simile a questo:

```
[
  [1. 0.07040032 0. 0.]
  [0.07040032 1. 0.07673165 0.]
  [0. 0. 0.02332112 0.]
]
```

In questa matrice bidimensionale, ogni riga riflette un fotogramma dello spazio di parcheggio. E ogni colonna indica quanto fortemente ciascuno dei luoghi si interseca con una delle macchine rilevate. Un risultato di 1.0 significa che l'intero spazio è interamente occupato dall'auto, e un valore basso come 0.02 indica che l'auto è salita un po' in posizione, ma puoi ancora parcheggiarci sopra.

Per trovare posti non occupati, devi solo controllare ogni riga in questo array. Se tutti i numeri sono vicini allo zero, molto probabilmente il posto è libero!

Tuttavia, tieni presente che il riconoscimento degli oggetti non funziona sempre correttamente con i video in tempo reale. Sebbene il modello basato su Mask R-CNN sia del tutto accurato, di tanto in tanto potrebbe mancare un'auto o due in un fotogramma del video. Pertanto, prima di affermare che il luogo è libero, è necessario assicurarsi che rimanga tale per i prossimi 5-10 fotogrammi successivi del video. In questo modo, possiamo evitare situazioni in cui il sistema contrassegna erroneamente un posto vuoto a causa di un problema tecnico in un fotogramma del video. Non appena ci assicuriamo che il posto rimanga libero per diversi frame, puoi inviare un messaggio!

Inviare SMS

L'ultima parte del nostro trasportatore sta inviando notifiche SMS quando appare un parcheggio gratuito.

Inviare un messaggio da Python è molto semplice se usi Twilio. Twilio è un'API accessibile che ti consente di inviare SMS da quasi tutti i linguaggi di programmazione con poche righe di codice. Naturalmente, se preferisci un servizio diverso, puoi utilizzarlo. Non ho niente a che fare con Twilio: è solo la prima cosa che mi viene in mente.

Per utilizzare Twilio, registrati per un account di prova, crea un numero di telefono Twilio e ottieni le informazioni di autenticazione dell'account. Quindi installa la libreria client:

```
$ pip3 install twilio
```

Successivamente, utilizza il seguente codice per inviare il messaggio:

```
from twilio.rest import Client

# Twilio account details.
twilio_account_sid = 'Your Twilio SID'
twilio_auth_token = 'Your Twilio Authentication Token'
twilio_source_phone_number = 'Your Twilio Phone Number'

# Create a Twilio client object.
client = Client(twilio_account_sid, twilio_auth_token)

# Send SMS.
message = client.messages.create (
    body = "Message body",
    from_ = twilio_source_phone_number,
    to = "Your number where the message will come"
)
```

Per aggiungere la possibilità di inviare messaggi al nostro script, copia lì questo codice. Tuttavia, è necessario assicurarsi che il messaggio non venga inviato su ogni frame,

dove puoi vedere lo spazio libero. Pertanto, avremo un flag che nello stato installato non consentirà l'invio di messaggi per un po 'di tempo o fino a quando un altro luogo non sarà libero.

Mettere tutto insieme

```

from mrcnn.model import MaskRCNN
from pathlib import Path
from twilio.rest import Client

# The configuration that the Mask-RCNN library will use.
class MaskRCNNConfig (mrcnn.config.Config):
    NAME = "coco_pretrained_model_config"
    IMAGES_PER_GPU = 1
    GPU_COUNT = 1
    NUM_CLASSES = 1 + 80 # in the COCO dataset there are 80 classes + 1 background class.
    DETECTION_MIN_CONFIDENCE = 0.6

# We filter the list of recognition results so that only cars remain.
def get_car_boxes (boxes, class_ids):
    car_boxes = []

    for i, box in enumerate (boxes):
        # If the found object is not a car, then skip it.
        if class_ids [i] in [3, 8, 6]:
            car_boxes.append (box)

    return np.array (car_boxes)

# Twilio configuration.
twilio_account_sid = 'Your Twilio SID'
twilio_auth_token = 'Your Twilio Authentication Token'
twilio_phone_number = 'Your Twilio Phone Number'
destination_phone_number = 'Number where the message will come'
client = Client (twilio_account_sid, twilio_auth_token)

# The root directory of the project.
ROOT_DIR = Path (".")

# Directory for saving logs and trained model.
MODEL_DIR = ROOT_DIR / "logs"

# Local path to the file with trained weights.
COCO_MODEL_PATH = ROOT_DIR / "mask_rcnn_coco.h5"

# Download COCO dataset if necessary.
if not COCO_MODEL_PATH.exists ():
    mrcnn.utils.download_trained_weights (COCO_MODEL_PATH)

# Directory with images for processing.
IMAGE_DIR = ROOT_DIR / "images"

# Video file or camera for processing - insert the value 0 if using a camera, not a video file.
VIDEO_SOURCE = "test_images / parking.mp4"

# Create a Mask-RCNN model in output mode.
model = MaskRCNN (mode = "inference", model_dir = MODEL_DIR, config = MaskRCNNConfig ())

# Download the pre-trained model.
model.load_weights (COCO_MODEL_PATH, by_name = True)

```

```
import numpy as np
import cv2
import mrcnn.config
import mrcnn.utils
```

Posizione dei parcheggi.

parked_car_boxes= None

Download the `video file` for which we want to run `recDgM` (i.e. `video_capture = cv2.VideoCapture(FIDE_O_SOURCE)`)

How many frames in a row `math` an `mnpR` place we have already seen.
`free_space_frames = 0`

Have we already sent SMS?
`sms_sent = False`

Wait for the `each` frame.
`while cv2.VideoCapture.isOpened():`
 `success, frame = video_capture.read()`
 if not success:
 break

Convert the image from the `BGR` color model to `RGB`.
`rgb_image = frame[..., ::-1]`

Use the `image` of the `flask R-CNN` model to get the result.
`results = mDdel.detect([rgb_image], srbDse=0)`

`flask R-CA` assumes that we recognize objects in multiple images.
If it is trained only on one image, we extract `Ofill` the first result.
`r = results[0]`

The variable `now` contains recognition results.
- `r.rois` - bounding box for each recognized object:
- `r.class_ids` - identifier (type) of the object:
- `r.score` - degree of confidence:
- `r.masks` - masks of objects (high gray values on their outline).

if `parked_car_boxes` is None:
 This is the first frame of the video - let's say that all detected cars are in the parking lot.
 Save the location of each car as a parking space and move on to the next frame.
 `parked_car_boxes = get_car_boxes([r.rois], r.class_ids)`
else:
 We already know where the places are. Check if there are any.

We are looking for `KOfSD` on the output frame.
`car_boxes = get_car_boxes([r.rois], r.class_ids)`

Let's check if these cars intersect with the known parking spaces.
`DSerlaps = rrcnn.utils.compute_overlap(parked_car_boxes, car_boxes)`

We assume that there are no empty seats until we find at least one.
`free_space = False`

We go through the cycle for each well-known parking space.
for parking area, overlap_areas in zip(parked_car_boxes, overlaps):


```

        = if there is a DH found for the maximum size of the intersection and detected
        On the frame b5 the machine (no matter which).
        max_IoU_overlap = np.max(overlap_areas)
# We get the upper left and lower right coordinates of the parking space.
y1, x1, y2, x2 = parking_area

# Check if space is free by checking the IoU value.
if max_IoU_overlap < 0.15:
    # Place is free! Draw a green frame around it.
    cv2.rectangle(frame, (x1, y1), (x2, y2), (0, 255, 0), 3)
    # We note that we found at least it is free space.
    free_space = True
else:
    # The place is still taken - we draw a red frame.
    cv2.rectangle(frame, (x1, y1), (x2, y2), (0, 0, 255), 1)

# Write the IoU value inside the frame.
font = cv2.FONT_HERSHEY_DUPLEX
cv2.putText(frame, f" {max_IoU_overlap: 0.2}", (x1 + 6, y2 - 6), font, 0.3, (255, 255, 255))

# If at least one place was free, we begin to count frames.
# This is to make sure that the place is really free
# and do not send another notification.
if free_space:
    free_space_frames += 1
else:
    # If everything is busy, reset the counter.
    free_space_frames = 0

# If the place is free for several frames, we can say that it is free.
if free_space_frames > 10:
    # Display SPACE AVAILABLE !! at the top of the screen.
    font = cv2.FONT_HERSHEY_DUPLEX
    cv2.putText(frame, f"SPACE AVAILABLE!", (10, 150), font, 3.0, (0, 255, 0), 2, cv2.FILLED)

# Send a message if you have not done so already.
if not sms_sent:
    print("SENDING SMS !!!")
    message = client.messages.create(
        body = "Parking space open - go go go!",
        from_ = twilio_phone_number,
        to = destination_phone_number
    )
    sms_sent = True

# Show the frame on the screen.
cv2.imshow('Video', frame)

# Press 'q' to exit.
if cv2.waitKey(1) & 0xFF == ord('q'):
    break

# Press 'q' to exit.
video_capture.release()
cv2.destroyAllWindows()

```

Per eseguire quel codice, devi prima installare Python 3.6+, [Maschera Matterport R-CNN](#), e [OpenCV](#).

Ho scritto specificamente il codice nel modo più semplice possibile. Ad esempio, se vede nel primo fotogramma dell'auto, conclude che sono tutte parcheggiate. Prova a sperimentarlo e vedi se riesci a migliorarne l'affidabilità.

Semplicemente modificando gli identificatori degli oggetti che il modello sta cercando, puoi trasformare il codice in qualcosa di completamente diverso. Ad esempio, immagina di lavorare in una stazione sciistica. Dopo aver apportato un paio di modifiche, puoi trasformare questo script in un sistema che riconosce automaticamente gli snowboarder che saltano da una rampa e registra video con salti fantastici. Oppure, se lavori in una riserva naturale, puoi creare un sistema che conta le zebre. Sei limitato solo dalla tua immaginazione.

7. CREAZIONE DI GIOCHI SULLA STRUTTURA PYGAME | PARTE 1



Ciao, amante di Python!

T questo è il primo di un tutorial in tre parti sulla creazione di giochi utilizzando Python 3 e Pygame. Nella seconda parte, abbiamo esaminato la classe `TextObject`, utilizzata per rendere il testo sullo schermo creato nella finestra principale, e abbiamo imparato a disegnare oggetti: mattoni, palla e racchetta.

In questa parte, ci immergeremo più a fondo nel cuore di Breakout e impareremo come gestire gli eventi, conosceremo la classe principale di Breakout e vedremo come spostare vari oggetti nel gioco.

Gestione degli eventi

Breakout ha tre tipi di eventi: eventi di battitura, eventi del mouse ed eventi timer. Il ciclo principale nella classe `Game` gestisce le sequenze di tasti e gli eventi del mouse e li passa ai sottoscrittori (chiamando una funzione del gestore).

Sebbene la classe `Game` sia molto generale e non conosca l'implementazione di Breakout, i metodi di gestione dell'abbonamento e degli eventi sono molto specifici.

classe breakout

La classe `Breakout` implementa la maggior parte delle conoscenze su come viene controllato Breakout. In questa serie di tutorial, incontreremo più volte la classe `Breakout`. Ecco le righe che registrano i vari gestori di eventi.

Va notato che tutti gli eventi chiave (per entrambe le "frecce" sinistra e destra) vengono trasmessi a un metodo di gestione della racchetta.

```
# Register the handle_mouse_event () method of the button object
self.mouse_handlers.append (b.handle_mouse_event)

# Register racket handle () method for handling key events
self.keydown_handlers [pygame.K_LEFT] .append (paddle.handle)
self.keydown_handlers [pygame.K_RIGHT] .append (paddle.handle)
self.keyup_handlers [pygame.K_LEFT] .append (paddle.handle)
self.keyup_handlers [pygame.K_RIGHT] .append (paddle.handle)
```

Gestione della sequenza di tasti

La classe Game chiama gestori registrati per ogni evento chiave e passa la chiave. Nota che questa non è una pagaia

classe. In Breakout, l'unico oggetto interessato a tali eventi è un racket. Quando si preme o si rilascia un tasto, viene chiamato il suo metodo handle(). L'oggetto Paddle non ha bisogno di sapere se questo è stato un evento di pressione o rilascio di un tasto, perché controlla lo stato corrente utilizzando una coppia di variabili booleane: Moving_left e Moving_right . Se move_left True, significa che è stato premuto il tasto "left", e l'evento successivo sarà il rilascio del tasto, che ripristinerà la variabile. Lo stesso vale per la chiave giusta. La logica è semplice e consiste nel commutare lo stato di queste variabili in risposta a qualsiasi evento.

```
def handle (self, key):  
    if key == pygame.K_LEFT:  
        self.moving_left = not self.moving_left  
    else:  
        self.moving_right = not self.moving_right
```

Gestione degli eventi del mouse

Breakout ha un menu di gioco che incontreremo presto. Il pulsante del menu controlla vari eventi del mouse, come il movimento e la pressione dei pulsanti (eventi mouse giù e mouse su). In risposta a questi eventi, il pulsante aggiorna la variabile di stato interna. Ecco il codice di elaborazione del mouse:

```

def handle_mouse_event (self, type, pos):
    if type == pygame.MOUSEMOTION:
        self.handle_mouse_move (pos)
    elif type == pygame.MOUSEBUTTONDOWN:
        self.handle_mouse_down (pos)
    elif type == pygame.MOUSEBUTTONUP:
        self.handle_mouse_up (pos)

def handle_mouse_move (self, pos):
    if self.bounds.collidepoint (pos):
        if self.state != 'pressed':
            self.state = 'hover'
    else:
        self.state = 'normal'

def handle_mouse_down (self, pos):
    if self.bounds.collidepoint (pos):
        self.state = 'pressed'

def handle_mouse_up (self, pos):
    if self.state == 'pressed':
        self.on_click (self)
        self.state = 'hover'

```

Si noti che il metodo `handle_mouse_event()` registrato per ricevere gli eventi del mouse controlla il tipo di evento e lo reindirizza al metodo appropriato che elabora questo tipo di evento.

Gestione degli eventi del timer

Gli eventi del timer non vengono elaborati nel ciclo principale. Tuttavia, poiché il ciclo principale viene chiamato in ogni frame, è facile verificare se è giunto il momento per un particolare evento. Lo vedrai più avanti quando parleremo degli effetti speciali temporanei.

Un'altra situazione è la necessità di mettere in pausa il gioco. Ad esempio, quando viene visualizzato un messaggio che il giocatore deve leggere e in modo che nulla lo distraiga. Il metodo della classe `show_message()` Breakout adotta questo approccio e chiama `time.sleep()`. Ecco il relativo codice:

```

import config as c

class Breakout (Game):
    def show_message (self,
        text
        color = colors.WHITE,
        font_name = 'Arial',
        font_size = 20,
        centralized = False):
        message = TextObject (c.screen_width // 2,
            c.screen_height // 2,
            lambda: text, color,
            font_name, font_size)
        self.draw ()
        message.draw (self.surface, centralized)
        pygame.display.update ()
        time.sleep (c.message_duration)

```

Processo di gioco

Il gameplay (gameplay) è il punto in cui entrano in gioco le regole di Breakout. Il gameplay consiste nello spostare vari oggetti in risposta agli eventi e nel cambiare lo stato del gioco in base alle loro interazioni.

Racchetta in movimento

Hai visto prima che la classe Paddle risponde ai tasti freccia aggiornando SUO

campi Moving_left e Moving_right . Il movimento stesso avviene in prossimità del metodo update(). Alcuni calcoli vengono eseguiti qui se la racchetta è bordo sinistro o destro dello schermo. Non vogliamo che la racchetta se ne vada oltre i confini dello schermo (tenendo conto di un dato offset).

Pertanto, se il movimento sposta l'oggetto oltre i bordi, l'unico movimento in il codice regola il movimento in modo che si fermi proprio al confine. Dal momento che la racchetta zero. orizzontale rispetto alla componente verticale del movimento è sempre

```

import pygame

import config as c
from game_object import GameObject

class Paddle(GameObject):
    def __init__(self, x, y, w, h, color, offset):
        GameObject.__init__(self, x, y, w, h)
        self.color = color
        self.offset = offset
        self.moving_left = False
        self.moving_right = False

    ...

    def update(self):
        if self.moving_left:
            dx = - (min (self.offset, self.left))
        elif self.moving_right:
            dx = min (self.offset, c.screen_width - self.right)
        else:
            return

        self.move (dx, 0)

```

Palla in movimento

La palla utilizza semplicemente la funzionalità della classe base `GameObject`, che sposta gli oggetti in base alla loro velocità (le sue componenti orizzontali e verticali). Come vedremo presto, la velocità di una palla è determinata da molti fattori nella classe `Breakout`. Poiché il movimento consiste semplicemente nell'aggiungere velocità alla posizione corrente, la direzione in cui si muove la palla è completamente determinata dalla velocità lungo gli assi orizzontale e verticale.

Impostazione della velocità iniziale della palla

La palla `Breakout` esce dal nulla all'inizio del gioco ogni volta che un giocatore perde la vita. Si materializza semplicemente dall'etere e inizia a cadere esattamente verso il basso o con una leggera angolazione. Quando la palla viene creata nel metodo `create_ball()`, ottiene la velocità con un componente orizzontale casuale nell'intervallo da - 2 a 2 e il componente verticale specificato nel modulo `config.py` (il valore predefinito è 3).

```
def create_ball(self):
    speed = (random.randint(-2, 2), c.ball_speed)
    self.ball = Ball(c.screen_width // 2,
                    c.screen_height // 2,
                    c.ball_radius,
                    c.ball_color,
                    speed)
    self.objects.append(self.ball)
```

Ricapitolare

In questa parte, abbiamo esaminato la gestione di eventi quali sequenze di tasti, movimenti del mouse e clic del mouse. Abbiamo anche esaminato alcuni elementi del gameplay di Breakout: spostare la racchetta, spostare la palla e controllare la velocità della palla.

Nella quarta parte, considereremo l'importante argomento del riconoscimento delle collisioni e vedremo cosa succede quando la palla colpisce diversi oggetti di gioco: una racchetta, mattoni, pareti, soffitto e pavimento. Quindi presteremo attenzione al menu di gioco. Creeremo i nostri pulsanti, che utilizzeremo come menu, e potremo mostrarli e nasconderli se necessario.

CREARE GIOCHI SU PYGAME

QUADRO | PARTE 2

Questo è il secondo di un tutorial in tre parti sulla creazione di giochi utilizzando Python 3 e Pygame. Nella terza parte, ci siamo addentrati nel cuore di Breakout e abbiamo imparato a gestire gli eventi, abbiamo fatto conoscenza con la classe principale di Breakout e abbiamo visto come spostare diversi oggetti di gioco.

In questa parte impareremo come riconoscere le collisioni e cosa succede quando una palla colpisce un oggetto diverso: una racchetta, mattoni, pareti, soffitto e pavimento. Infine, esamineremo l'importante argomento dell'interfaccia utente e, in particolare, come creare il menu dai pulsanti.

Riconoscimento collisione

Nei giochi, gli oggetti si scontrano tra loro e Breakout non fa eccezione. La palla si scontra con gli oggetti. Il metodo principale `handle_ball_collisions()` ha una funzione incorporata chiamata `intersect()` che viene utilizzata per verificare se la palla ha colpito l'oggetto e dove si è scontrata con l'oggetto. Restituisce "sinistra", "destra", "alto", "basso" o Nessuno se la palla non lo fa

scontrarsi con un oggetto.

```
def handle_ball_collisions(self):
    def intersect(obj, ball):
        edges = dict (
            left = Rect (obj.left, obj.top, 1, obj.height),
            right = Rect (obj.right, obj.top, 1, obj.height),
            top = Rect (obj.left, obj.top, obj.width, 1),
            bottom = Rect (obj.left, obj.bottom, obj.width, 1))
        collisions = set (edge for edge, rect in edges.items () if
            ball.bounds.colliderect (rect))
        if not collisions:
            return none

        if len (collisions) == 1:
            return list (collisions) [0]

        if 'top' in collisions:
            if ball.centery >= obj.top:
                return 'top'
            if ball.centerx < obj.left:
                return 'left'
            else:
                return 'right'

        if 'bottom' in collisions:
            if ball.centery >= obj.bottom:
                return 'bottom'
            if ball.centerx < obj.left:
                return 'left'
            else:
                return 'right'
```

Collisione di una palla con una racchetta.

Quando la palla colpisce la racchetta, rimbalza. Se colpisce la parte superiore della racchetta, rimbalza indietro, ma mantiene gli stessi componenti ad alta velocità orizzontale.

Ma se colpisce il lato della racchetta, rimbalza sul lato opposto (destra o sinistra) e continua a scendere fino a toccare il pavimento. Il codice utilizza una funzione intersect().

```
# Kick on the racket
s = self.ball.speed
edge = intersect (self.paddle, self.ball)
if edge is not None:
    self.sound_effects ['paddle_hit']. play ()
if edge == 'top':
    speed_x = s [0]
    speed_y = -s [1]
if self.paddle.moving_left:
    speed_x -= 1
elif self.paddle.moving_right:
    speed_x += 1
self.ball.speed = speed_x, speed_y
elif edge in ('left', 'right'):
    self.ball.speed = (-s [0], s [1])
```

Collisione con il pavimento.

La palla colpisce la racchetta di lato, la palla continua a cadere e poi colpisce il pavimento. In questo momento, il giocatore perde la vita e la palla viene ricreata in modo che il gioco possa continuare. Il gioco termina quando il giocatore esaurisce la vita.

```
# Hit the floor
if self.ball.top > c.screen_height:
    self.lives -= 1
if self.lives == 0:
    self.game_over = True
else:
    self.create_ball ()
```

Collisione con soffitto e pareti

Quando una palla colpisce il muro o il soffitto, rimbalza semplicemente su di essi.

```
# Hit the ceiling
if self.ball.top < 0:
    self.ball.speed = (s [0], -s [1])

# Kick against the wall
if self.ball.left < 0 or self.ball.right > c.screen_width:
    self.ball.speed = (-s [0], s [1])
```

Collisione con mattoni

Quando la palla colpisce un mattone, questo è l'evento principale del gioco Breakout: il mattone è scomparso, il giocatore riceve un punto, la palla rimbalza e si verificano molti altri eventi (effetto sonoro e talvolta un effetto speciale), che considereremo dopo.

Per determinare che la palla ha colpito un mattone, il codice controllerà se uno dei mattoni si interseca con la palla:

```
#Bump on a brick
for brick in self.bricks:
    edge = intersect (brick, self.ball)
    if not edge:
        continue

    self.bricks.remove (brick)
    self.objects.remove (brick)
    self.score += self.points_per_brick

    if edge in ('top', 'bottom'):
        self.ball.speed = (s [0], -s [1])
    else:
        self.ball.speed = (-s [0], s [1])
```

Programma di progettazione del menu di gioco

La maggior parte dei giochi ha una sorta di UI Breakout ha un menu semplice con due pulsanti, "PLAY" e "QUIT". Il menu viene visualizzato all'inizio del gioco e scompare quando il giocatore fa clic su "GIOCA".

Vediamo come vengono implementati il pulsante e il menu e come si integrano nel gioco.

Creazione pulsante

Pygame non ha una libreria dell'interfaccia utente integrata. Le estensioni di terze parti, ma per il menu, abbiamo deciso di creare i nostri pulsanti. che ha tre stati: normale, evidenziato e premere. Lo stato normale quando il mouse non è sopra i pulsanti e lo stato di evidenziazione è quando il mouse è sopra il pulsante, il pulsante sinistro del mouse non è ancora premuto. Lo stato di stampa è quando il mouse è sopra il pulsante e il giocatore ha premuto il pulsante sinistro del mouse.

I pulsanti si implementano come un rettangolo con un colore di sfondo e un display di testo sopra di esso. Il pulsante riceve anche la (funzione onclick), che viene chiamata quando si fa clic sul pulsante.

```

import pygame

from game_object import GameObject
from text_object import TextObject
import config as c

class Button (GameObject):
    def __init__ (self,
        x
        y
        w
        h
        text
        on_click = lambda x: None,
        padding = 0):
        super () . __init__ (x, y, w, h)
        self.state = 'normal'
        self.on_click = on_click

        self.text = TextObject (x + padding,
            y + padding, lambda: text,
            c.button_text_color,
            c.font_name,
            c.font_size)

    def draw (self, surface):
        pygame.draw.rect (surface,
            self.back_color,
            self.bounds)
        self.text.draw (surface)

```

Il pulsante elabora i propri eventi del mouse e cambia il proprio stato interno in base a questi eventi. Quando il pulsante è in stato di pressione e riceve l'evento MOUSE BUTTONUP , significa che il giocatore ha premuto il pulsante e la funzione viene chiamata su _ click().

```

def handle_mouse_event (self, type, pos):
    if type == pygame.MOUSEMOTION:
        self.handle_mouse_move (pos)
    elif type == pygame.MOUSEBUTTONDOWN:
        self.handle_mouse_down (pos)
    elif type == pygame.MOUSEBUTTONUP:
        self.handle_mouse_up (pos)

def handle_mouse_move (self, pos):
    if self.bounds.collidepoint (pos):
        if self.state != 'pressed':
            self.state = 'hover'
    else:
        self.state = 'normal'

def handle_mouse_down (self, pos):
    if self.bounds.collidepoint (pos):
        self.state = 'pressed'

def handle_mouse_up (self, pos):
    if self.state == 'pressed':
        self.on_click (self)
        self.state = 'hover'

```

La proprietà `back_color` utilizzata per disegnare sul rettangolo di sfondo restituisce sempre il colore corrispondente alla forma corrente del pulsante, in modo che sia chiaro al giocatore che il pulsante è attivo:

```

@property
def back_color (self):
    return dict (normal = c.button_normal_back_color,
                hover = c.button_hover_back_color,
                pressed = c.button_pressed_back_color) [self.state]

```

Menu Design

La funzione `create_menu()` crea un menu con due pulsanti con il testo 'PLAY' e 'QUIT.' Ha due funzioni integrate, su `_play()` e su `_quit()` che passa al pulsante corrispondente. Ogni pulsante viene aggiunto all'elenco degli oggetti (per il rendering), così come nel menu dei campi `_pulsanti`.

```

def create_menu (self):
    for i, (text, handler) in enumerate (((('PLAY', on_play),
                                           ('QUIT', on_quit))):
        b = Button (c.menu_offset_x,
                    c.menu_offset_y + (c.menu_button_h + 5) * i,
                    c.menu_button_w,
                    c.menu_button_h,
                    text,
                    handler,
                    padding = 5)
        self.objects.append (b)
        self.menu_buttons.append (b)
        self.mouse_handlers.append (b.handle_mouse_event)

```

Quando PLAY il pulsante è presente onplay() , viene chiamata una funzione che rimuove il pulsante dall'oggetto elenco in modo che non sia più disegnato. In aggiunta, i valori del campo booleano che attivano l'inizio del gioco - is_gamerunning e startlevel - Va bene

Quando il pulsante viene premuto, QUIT è _ game_ running assume valore (False) (in effetti, mettendo in pausa il gioco), imposta game_ over su True, che attiva la sequenza di completamento del gioco.

```
def on_play (button):  
    for b in self.menu_buttons:  
        self.objects.remove (b)  
  
    self.is_game_running = True  
    self.start_level = True  
  
def on_quit (button):  
    self.game_over = True  
    self.is_game_running = False
```

Mostra e nascondi GameMenu

La visualizzazione e l'occultamento del menu vengono eseguiti in modo implicito. Quando i pulsanti sono nell'oggetto elenco, il menu è visibile: quando vengono rimossi, è nascosto. Tutto è molto semplice.

Crea un menu integrato con la sua superficie che rende i suoi sottocomponenti (pulsanti e altri oggetti) e quindi aggiungi o rimuovi semplicemente questi componenti di menu, ma questo non è richiesto per un menu così semplice.

Riassumere

Abbiamo esaminato il riconoscimento delle collisioni e cosa succede quando la palla si scontra con i diversi oggetti: una racchetta, mattoni, pareti, pavimento e soffitto. Abbiamo anche creato un menu con i nostri pulsanti, che possono essere nascosti e visualizzati a comando.

Nell'ultima parte della serie, prenderemo in considerazione il completamento del gioco, i punti di tracciamento e le vite, gli effetti sonori e la musica.

Sviluppiamo un complesso sistema di effetti speciali che aggiungono poche spezie al gioco. Infine, discuteremo di ulteriori sviluppi e possibili miglioramenti.

CREARE GIOCHI SU PYGAME

QUADRO | PARTE 3

Questa è l'ultima delle parti Third del tutorial sulla creazione di giochi utilizzando Python 3 e PyGame. Nella quarta parte, abbiamo imparato a riconoscere le collisioni, a rispondere al fatto che la palla si scontra con diversi oggetti di gioco e abbiamo creato un menu di gioco con i suoi pulsanti.

Nell'ultima parte, esamineremo vari argomenti: la fine del gioco, la gestione di vite e punti, gli effetti sonori, la musica e persino un sistema flessibile di effetti speciali. Per dessert, prenderemo in considerazione possibili miglioramenti e indicazioni per ulteriori sviluppi.

Fine del gioco

Prima o poi il gioco dovrebbe finire. In questa forma di Breakout, il gioco termina in due modi: il giocatore perde tutta la sua vita o distrugge tutti i mattoni. Non esiste un livello successivo nel gioco (ma può essere facilmente aggiunto).

Game Over!

La `game_over` della classe `Game` è impostata su `False` nel metodo `init()` della classe `Game`. Il ciclo principale continua fino a quando la variabile `game_over` changes in `True` :

```
class Game:
    def __init__(self,
                  caption,
                  width,
                  height,
                  back_image_filename,
                  frame_rate):
        ...
        self.game_over = False
        ...

    def run(self):
        while not self.game_over:
            self.surface.blit(self.background_image, (0, 0))

            self.handle_events()
            self.update()
            self.draw()

        pygame.display.update()
        self.clock.tick(self.frame_rate)
```

Tutto questo avviene nella classe `Breakout` nei seguenti casi:

- Il giocatore preme il pulsante QUIT nel menu. Il
- giocatore perde la sua ultima vita.
- Il giocatore distrugge tutti i mattoni.

```
def on_quit (button):
    self.game_over = True
    self.is_game_running = False

def handle_ball_collisions (self):
    ...
    # Hit the floor
    if self.ball.top > c.screen_height:
        self.lives -= 1
        if self.lives == 0:
            self.game_over = True

            if not self.bricks:
                self.show_message ('YOU WIN !!!', centralized = True)
                self.is_game_running = False
                self.game_over = True
            return

def update (self):
    ...
    if not self.bricks:
        self.show_message ('YOU WIN !!!', centralized = True)
        self.is_game_running = False
        self.game_over = True
    return
```

Display del messaggio di fine partita

Di solito, alla fine del gioco, non vogliamo che la finestra di gioco scompaia silenziosamente. Un'eccezione è un caso in cui si fa clic sul pulsante QUIT nel menu. Quando un giocatore perde la sua ultima vita, Breakout mostra il tradizionale messaggio 'GAME OVER!', e quando il giocatore vince, mostra il messaggio 'YOUWIN!'

In entrambi i casi, viene utilizzata la funzione `show_message()`. Visualizza il testo nella parte superiore della schermata corrente (il gioco si interrompe) e attende alcuni secondi prima di tornare. La successiva iterazione del ciclo di gioco, controllando il campo `game_over` determinerà che è `True`, dopodiché il programma terminerà.

Ecco come appare la funzione `show_message()`:

```

def show_message(self,
    text
    color = colors.WHITE,
    font_name = 'Arial',
    font_size = 20,
    centralized = False):
    message = TextObject(c.screen_width // 2,
        c.screen_height // 2,
        lambda text,
        color
        font_name
        font_size)
    self.draw()
    message.draw(self.surface, centralized)
    pygame.display.update()
    time.sleep(c.message_duration)

```

Salvataggio dei record tra i giochi

In questa versione del gioco, non salviamo i record, perché c'è un solo livello e i risultati di tutti i giocatori dopo la distruzione dei mattoni saranno gli stessi. In generale, il salvataggio dei record può essere implementato localmente, salvando i record in un file e visualizzando un altro messaggio se un giocatore supera un record.

Aggiunta di effetti sonori e musica

I giochi sono un processo audiovisivo. Molti giochi hanno effetti sonori: brevi clip audio che vengono riprodotti quando un giocatore uccide i mostri trova un tesoro o una morte terribile. Alcuni giochi hanno anche una musica di sottofondo che contribuisce all'atmosfera. Ci sono solo effetti sonori in Breakout, ma ti mostreremo come riprodurre la musica nei tuoi giochi.

Effetti sonori

Per riprodurre gli effetti sonori, abbiamo bisogno di file audio (come nel caso dei file di immagine). Questi file possono essere in formato .wav, .mp3 o .ogg. Breakout memorizza i suoi effetti sonori in una cartella `sound_effects`:

```

~/git/pygame-breakout> tree sound_effects /
sound_effects/
+ - - brick_hit.wav
+ - - effect_done.wav
+ - - level_complete.wav
+ - - paddle_hit.wav

```

Vediamo come questi effetti sonori vengono caricati e riprodotti al momento giusto. Per prima cosa, riproduci gli effetti sonori (o la musica di sottofondo), dobbiamo inizializzare il sistema audio Pygame. Questo viene fatto nella classe `Game:pygame.mixer.pre_init(44100, 16, 2,`

4096)

Quindi, nella classe Breakout, tutti gli effetti sonori vengono caricati da config nell'oggetto pygame mixer Sound e memorizzati nel dizionario:

```
# In config.py
sounds_effects = dict (
    brick_hit = 'sound_effects / brick_hit.wav',
    effect_done = 'sound_effects / effect_done.wav',
    paddle_hit = 'sound_effects / paddle_hit.wav',
    level_complete = 'sound_effects / level_complete.wav',
)

# In breakout.py
class Breakout (Game):
    def __init__ (self):
        ...
        self.sound_effects = {
            name: pygame.mixer.Sound (sound)
            for name, sound in c.sounds_effects.items ()
        }
        ...
```

Ora possiamo riprodurre effetti sonori quando succede qualcosa di interessante. Ad esempio, quando una palla colpisce un mattone:

```
#Bump on a brick
for brick in self.bricks:
    edge = intersect (brick, self.ball)
    if not edge:
        continue

self.sound_effects ['brick_hit']. play ()
```

L'effetto sonoro viene riprodotto in modo asincrono: ovvero, il gioco non si ferma durante la riproduzione. È possibile riprodurre più effetti sonori contemporaneamente.

Registra i tuoi effetti sonori e messaggi

Registrare i tuoi effetti sonori può essere un'esperienza semplice e divertente. A differenza della creazione di risorse visive, non richiede molto talento. Chiunque può dire "Boom!" o "Salta" o grida: "Ti hanno ucciso. Sii fortunato un'altra volta! "

Riproduzione di musica di sottofondo

La musica di sottofondo dovrebbe essere riprodotta continuamente. In teoria, è possibile creare un effetto sonoro molto lungo, ma la musica di sottofondo in loop viene utilizzata più spesso. I file musicali possono essere in formato .wav, .mp3 o .midi. Ecco come viene implementata la musica:

```
music = pygame.mixer.music.load ('background_music.mp3')
pygame.mixer.music.play (-1, 0.0)
```


È possibile riprodurre solo una musica di sottofondo alla volta. Tuttavia, diversi effetti sonori possono essere riprodotti sulla musica di sottofondo. Questo è ciò che si chiama miscelazione.

Aggiunta di funzionalità avanzate

Facciamo qualcosa di curioso. È interessante distruggere i mattoni con una palla, ma dà subito fastidio. E il sistema generale degli effetti speciali? Svilupperemo un sistema estensibile di effetti speciali associato ad alcuni mattoncini, che si attiva quando la pallina colpisce il mattoncino.

Questo sarà il piano. Gli effetti hanno una vita. L'effetto inizia quando il mattone crolla e termina quando l'effetto scade. Cosa succede se la palla colpisce un altro mattone con un effetto speciale? In teoria, puoi creare effetti compatibili, ma per semplificare tutto nell'implementazione originale, l'effetto attivo si fermerà e un nuovo effetto prenderà il suo posto.,

Sistema di effetti speciali

Nel caso più generale, un effetto speciale può essere definito come due scopi. Il primo ruolo attiva l'effetto e il secondo lo resetta. Vogliamo applicare effetti ai mattoncini e dare al giocatore una chiara comprensione di quali mattoncini speciali, in modo che possano provare a colpirli o evitarli in determinati punti.

I nostri effetti speciali sono determinati dal dizionario del modulo `breakout.py`. Ogni effetto ha un nome (ad esempio, `long_paddle`) e un valore che consiste in un colore mattone, oltre a due funzioni. Le funzioni sono definite funzioni lambda che prendono un'istanza di gioco, che include tutto ciò che può modificare l'effetto speciale in Breakout.

```

special_effects = dict (
    long_paddle = (
        colors.ORANGE,
        lambda g: g.paddle.bounds.inflate_ip (
            c.paddle_width // 2, 0),
        lambda g: g.paddle.bounds.inflate_ip (
            -c.paddle_width // 2, 0)),
    slow_ball = (
        colors.AQUAMARINE2,
        lambda g: g.change_ball_speed (-1),
        lambda g: g.change_ball_speed (1)),
    tripple_points = (
        colors.DARKSEAGREEN4,
        lambda g: g.set_points_per_brick (3),
        lambda g: g.set_points_per_brick (1)),
    extra_life = (
        colors.GOLD1,
        lambda g: g.add_life (),
        lambda g: None))

```

Quando si creano mattoni, è possibile assegnare loro uno degli effetti speciali. Ecco il codice:

```

def create_bricks (self):
    w = c.brick_width
    h = c.brick_height
    brick_count = c.screen_width // (w + 1)
    offset_x = (c.screen_width - brick_count * (w + 1)) // 2

    bricks = []
    for row in range (c.row_count):
        for col in range (brick_count):
            effect = None
            brick_color = c.brick_color
            index = random.randint (0, 10)
            if index < len (special_effects):
                x = list (special_effects.values ()) [index]
                brick_color = x [0]
                effect = x [1:]

            brick = Brick (offset_x + col * (w + 1),
                          c.offset_y + row * (h + 1),
                          w,
                          h,
                          brick_color,
                          effect)
            bricks.append (brick)
            self.objects.append (brick)
    self.bricks = bricks

```

The Brick class has an

effetto, che di solito ha il valore Nessuno, ma (con una probabilità del 30%) può contenere uno degli effetti speciali sopra definiti. Nota che questo codice non sa quali effetti esistono. Riceve semplicemente l'effetto e il colore del mattone e, se necessario, li assegna.

In questa versione di Breakout, attiviamo gli effetti solo quando colpiamo un mattone, ma puoi trovare altre opzioni per attivare gli eventi. L'effetto precedente viene scartato (se esisteva) e quindi viene lanciato un nuovo effetto. La funzione di ripristino e l'ora di inizio dell'effetto vengono memorizzate per un uso futuro.

```
if brick.special_effect is not None:
    # Reset the previous effect, if any
    if self.reset_effect is not None:
        self.reset_effect(self)

    # Triggering a special effect
    self.effect_start_time = datetime.now()
    brick.special_effect[0](self)
    # Setting the current effect reset function
    self.reset_effect = brick.special_effect[1]
```

Se il nuovo effetto non viene lanciato, dobbiamo comunque ripristinare l'effetto corrente dopo la sua durata. Questo accade nel metodo update(). In ogni frame, al campo reset_effect è assegnata una funzione per ripristinare l'effetto corrente. Se il tempo dopo l'avvio dell'effetto corrente supera la durata dell'effetto, la funzione si chiama reset_effect() e il campo reset_effect assume il valore None (nel senso che non ci sono effetti attivi al momento).

```
# Reset special effect if necessary
if self.reset_effect:
    elapsed = datetime.now() - self.effect_start_time
    if elapsed >= timedelta(seconds=c.effect_duration):
        self.reset_effect(self)
        self.reset_effect = None
```

Aumento della racchetta

L'effetto di una racchetta lunga è di aumentare la racchetta del 50%. La sua funzione di reset riporta la racchetta alle sue dimensioni normali. Il mattone ha il colore Arancio.:

```
long_paddle = (
    colors.ORANGE,
    lambda g: g.paddle.bounds.inflate_ip(
        c.paddle_width // 2, 0),
    lambda g: g.paddle.bounds.inflate_ip(
        -c.paddle_width // 2, 0),
```

Rallentamento della palla

Un altro effetto che aiuta a inseguire la palla è il rallentamento della palla, ovvero la riduzione della sua velocità di un'unità. Il mattone ha un colore Acquamarina.

```
slow_ball = (colors.AQUAMARINE2,
    lambda g: g.change_ball_speed(-1),
    lambda g: g.change_ball_speed(1)),
```

Più punti

Se vuoi grandi risultati, allora ti piacerà l'effetto di triplicare i punti, dando tre punti per ogni mattone distrutto invece del punto standard. Il mattone è verde scuro.

```
triple_points = (colors.DARKSEAGREEN4,  
                 lambda g: g.set_points_per_brick (3),  
                 lambda g: g.set_points_per_brick (1)),
```

Vite extra

Infine, un effetto molto utile sarà l'effetto di vite extra. Ti dà solo un'altra vita. Non ha bisogno di un reset. Il mattone ha un colore dorato.

```
extra_life = (colors.GOLD1,  
              lambda g: g.add_life (),  
              lambda g: None))
```

Funzioni future

Esistono diverse direzioni logiche per espandere Breakout. Se sei interessato a provare te stesso ad aggiungere nuove caratteristiche e funzioni, ecco alcune idee.

Passa al livello successivo

Per trasformare Breakout in un gioco serio, hai bisogno di livelli: uno non basta. All'inizio di ogni livello, ripristineremo lo schermo, ma salveremo punti e vite. Per complicare il gioco, puoi aumentare leggermente la velocità della palla ad ogni livello o aggiungere un altro strato di mattoni.

Seconda palla

L'effetto dell'aggiunta temporanea di una seconda palla creerà un enorme caos. La difficoltà qui è trattare entrambe le palle come uguali, indipendentemente da quale sia l'originale. Quando una pallina scompare, il gioco continua con l'unica rimasta. La vita non è persa.

Record duraturi

Quando hai livelli con difficoltà crescente, è consigliabile creare una tabella dei punteggi più alti. Puoi memorizzare i record in un file in modo che vengano salvati dopo la partita. Quando un giocatore batte un record, puoi aggiungere piccole pizze o fargli scrivere un nome (tradizionalmente con solo tre caratteri).

Bombe e bonus

Nell'attuale implementazione in poi, tutti gli effetti speciali sono associati ai mattoni, ma puoi aggiungere effetti (buoni e cattivi) che cadono dal cielo, che il giocatore può raccogliere o evitare.

Ricapitolare

Lo sviluppo di Breakout con Python 3 e Pygame si è rivelato un'esperienza piacevole. Questa è una combinazione avvincente per creare giochi 2D (e anche per giochi 3D). Se ami Python e vuoi creare i tuoi giochi, allora non esitare a scegliere Pygame.

8. PROGRAMMAZIONE ORIENTATA AGLI OGGETTI (OOP) IN PYTHON 3

Algoritmi e strutture dati

Ciao, amante di Python!

Sommario

- Che cos'è la programmazione orientata agli oggetti (OOP)?
- classi Python
- Oggetto Python (istanze)
- Come definire la classe in Python Instance Attribute Class Attributes
- Creazione di oggetti Cos'è? Panoramica dell'esercizio (# 1)
- Metodi di istanza Modifica degli attributi
- Ereditarietà degli oggetti Python Esempio di un parco per cani Espansione della funzionalità della classe padre Classi padre e classi figlio Override della funzionalità della classe padre Panoramica dell'esercizio (# 2)
- Conclusione

acquisirai familiarità con i seguenti concetti di base dell'OOP in Python:

- Classi Python
- Istanze di oggetti
- Definizione e lavoro con metodi
- OOP Ereditarietà

Che cos'è la programmazione orientata agli oggetti (OOP)?

La programmazione orientata agli oggetti, o, in breve, OOP, è una [paradigma di programmazione](#) che fornisce un mezzo per strutturare i programmi in modo tale che proprietà e comportamento siano combinati in oggetti separati.

Ad esempio, un oggetto può rappresentare una persona con un nome, età, indirizzo, ecc., con comportamenti come camminare, parlare, respirare e correre. Oppure un'e-mail con proprietà come elenco di destinatari, oggetto, corpo, ecc., nonché con comportamenti come l'aggiunta di allegati e l'invio.

In altre parole, la programmazione orientata agli oggetti è un approccio per modellare oggetti reali specifici, come le automobili, nonché le relazioni tra cose come aziende e dipendenti, studenti e insegnanti, ecc. OOP modella oggetti reali come oggetti di programma che hanno alcuni dati che sono ad esso associati e possono svolgere determinate funzioni.

Un altro paradigma di programma comune è il programma procedurale, che struttura un programma come una ricetta, nel senso che fornisce una serie di passaggi sotto forma di funzioni e blocchi di codice che vengono eseguiti in sequenza per completare un'attività.

La conclusione chiave è che gli oggetti sono al centro del paradigma della programmazione orientata agli oggetti, rappresentando non solo i dati, come nella programmazione procedurale, ma anche la struttura generale del programma.

NOTA Dal momento che Python è un linguaggio di programmazione con molti paradigmi, puoi scegliere il paradigma più adatto al problema in questione, mescolare diversi paradigmi in un programma o passare da un paradigma all'altro man mano che il tuo programma si sviluppa.

classi Python

Concentrandosi prima sui dati, ogni oggetto o cosa è un'istanza di una classe.

Le strutture dati primitive disponibili in Python, come numeri, stringhe ed elenchi, sono progettate per rappresentare cose semplici, come il valore di qualcosa, il nome della poesia e i tuoi colori preferiti, rispettivamente.

E se volessi immaginare qualcosa di molto più complesso?

Ad esempio, supponiamo di voler tracciare diversi animali. Se usi una lista, il primo elemento può essere il nome dell'animale, mentre il secondo elemento può rappresentare la sua età.

Come fai a sapere quale elemento dovrebbe essere? E se avessi 100 animali diversi? Sei sicuro che ogni animale abbia un nome, un'età e così via? E se volessi aggiungere altre proprietà a questi animali? Questa non è abbastanza organizzazione, ed è esattamente ciò di cui hai bisogno per le lezioni.

Le classi vengono utilizzate per creare una nuova struttura di dati utente che contiene arbitrari

informazioni su qualcosa. In questo caso di un animale, potremmo creare una classe `Animal()` per tenere traccia delle proprietà dell'animale come nome ed età.

È importante notare che una classe fornisce semplicemente una struttura: è un esempio di come dovrebbe essere definito qualcosa, ma in realtà non fornisce alcun contenuto reale. `Animal()` La classe può indicare che il nome e l'età sono necessari per determinare l'animale, ma non afferma che il nome o l'età di un particolare animale lo sia.

Questo può aiutare a presentare la classe come un'idea di come qualcosa dovrebbe essere definito.

Oggetti Python (istanze)

Mentre una classe è un piano, un'istanza è una copia di una classe con valori effettivi, letteralmente un oggetto appartenente a una particolare classe. Questa non è più un'idea: è un animale vero, come un cane di nome Roger, che ha otto anni.

In altre parole, una classe è un modulo o un profilo. Determina le informazioni necessarie. Dopo aver compilato il modulo, la tua copia specifica è un'istanza della classe: contiene informazioni aggiornate rilevanti per te.

Puoi compilare più copie per creare molte copie diverse, ma senza un modulo, come guida, ti perderesti senza sapere quali informazioni sono richieste. Quindi, prima di poter creare istanze separate di un oggetto, dobbiamo prima specificare di cosa hai bisogno definendo una classe.

Come definire una classe in Python

Definire una classe è semplice in Python:

```
class Dog:
    pass
```

Inizi con una `class` keyword per indicare che stai creando una classe, quindi aggiungi il nome della classe (usando [la notazione CamelCase](#) iniziando con la maiuscola).

Anche qui abbiamo usato la parola chiave Python `pass`. Questo è enorme spesso usato come segnaposto dove alla fine andrà il codice. Questo ci permette di eseguire questo codice senza generare un errore.

Nota: il codice sopra è corretto in Python 3. Su Python 2.x ("Python deprecato"), useresti una definizione di classe leggermente diversa:

```
# Python 2.x Class Definition:
class Dog (object):
    pass
```


Non le parti (oggetto) tra parentesi indicano la classe genitore da cui stai ereditando (più su questo sotto). In Python-3, questo non è più necessario perché è implicito per impostazione predefinita.

Attributo istanza

Tutte le classi creano oggetti e tutti gli oggetti contengono caratteristiche chiamate attributi (chiamate proprietà nel primo paragrafo). Utilizzare il metodo `init()` per inizializzare (ad esempio, determinare) gli attributi iniziali di un oggetto assegnando loro un valore predefinito (stato). Questo metodo deve avere almeno un argomento, oltre a una variabile `self` che fa riferimento all'oggetto stesso (ad esempio, Cane).

```
class Dog:
    # Initializer / Instance Attributes
    def __init__(self, name, age):
        self.name = name
        self.age = age
```

Nella nostra classe `Dog()`, ogni cane ha un nome e un'età specifici, che è sicuramente importante sapere quando inizi a creare cani diversi. Ricorda: la classe ha lo scopo solo di definire un cane e non di creare istanze di singoli cani con nomi ed età specifici: su questo torneremo presto.

Allo stesso modo, una variabile `self` è anche un'istanza di una classe. Poiché le istanze di classe hanno significati diversi, potremmo obiettare, `Dog.name = namenot` `self.name = name`. Ma poiché non tutti i cani hanno lo stesso nome, dobbiamo essere in grado di assegnare valori diversi per istanze diverse. Da qui la necessità di una variabile `self` speciale che aiuti a tenere traccia delle singole istanze di ogni classe.

NOTA: non dovrai mai chiamare un metodo `init()`: viene chiamato automaticamente quando viene creata una nuova istanza di `Dog`.

Attributi di classe

Sebbene gli attributi di istanza siano specifici per ogni oggetto, gli attributi di classe sono gli stessi per tutte le istanze, in questo caso, tutti i cani.

```
class Dog:
    # Class Attribute
    species = 'mammal'

    # Initializer / Instance Attributes
    def __init__(self, name, age):
        self.name = name
        self.age = age
```

Quindi, anche se ogni cane ha un nome e un'età unici, ogni cane sarà a

mammifero. Creiamo dei cani...

Crea oggetti

Istanziare è un termine insolito per la creazione di una nuova istanza univoca di una classe.

Ad esempio: >>>

```
>>> class Dog:
...     pass
...
>>> Dog ()
<__main__.Dog object at 0x1004ccc50>
>>> Dog ()
<__main__.Dog object at 0x1004ccc90>
>>> a = Dog ()
>>> b = Dog ()
>>> a == b
False
```

Abbiamo iniziato definendo una nuova classe Dog(), quindi abbiamo creato due nuovi cani, ognuno dei quali è stato assegnato a diversi

oggetti. Quindi, per creare un'istanza della classe, usi il nome della classe seguito da parentesi. Quindi, per dimostrare che ogni istanza è in realtà diversa, abbiamo creato altri due cani, assegnando ciascuna variabile e quindi verificando se queste variabili sono uguali.

Quale pensi sia il tipo di istanza di classe? >>>

```
>>> class Dog:
...     pass
...
>>> a = Dog ()
>>> type (a)
<class '__main__.Dog'>
```

Vediamo l'esempio più complesso...

```
class Dog:

    # Class Attribute
    species = 'mammal'

    # Initializer / Instance Attributes
    def __init__(self, name, age):
        self.name = name
        self.age = age

# Instantiate the Dog object
philo = Dog ("Philo", 5)
mikey = Dog ("Mikey", 6)

# Access the instance attributes
print ("{} is {} and {} is {}".format (
    philo.name, philo.age, mikey.name, mikey.age))

# Is Philo a mammal?
if philo.species == "mammal":
    print ("{} is a {}!".format (philo.name, philo.species))
```

NOTA Nota come utilizziamo i record di punti per accedere agli attributi di ciascun oggetto.
Salva con nome (dog_class.py), quindi esegui il programma. Tu dovresti vedere:

```
Philo is 5 and Mikey is 6.  
Philo is a mammal!
```

Che cosa c'è?

Creiamo una nuova istanza della classe Dog() e la assegniamo a una variabile Philo. Poi gli abbiamo passato due argomenti, "Philo" e 5, che rappresentano rispettivamente il nome e l'età di questo cane.

Questi attributi vengono passati al metodo init, che viene chiamato ogni volta che si crea un nuovo allegato, istanza il nome e l'età dell'oggetto. Forse ti starai chiedendo perché non avremmo dovuto dare argomenti a noi stessi.

Questa è la magia di Python: quando crei una nuova istanza della classe, Python determina automaticamente quali selfie (in questo caso, Dog) e li passa al metodo init.

Revisione degli esercizi (n. 1)

Esercizio: "Il cane più vecchio"

Usando la stessa Dogclass, crea tre nuovi cani, ognuno con un'età diversa. Quindi scrivi una funzione con un nome get_biggest_number() che richiede un numero qualsiasi di età (*args) e restituisce il più vecchio. Quindi stampa l'età del cane più vecchio in questo modo:

Il cane più vecchio ha 7 anni. Soluzione: Soluzione "Il cane più vecchio"
"Il cane più vecchio."

```

class Dog:

    # Class Attribute
    species = 'mammal'

    # Initializer / Instance Attributes
    def __init__(self, name, age):
        self.name = name
        self.age = age

# Instantiate the Dog object
jake = Dog ("Jake", 7)
doug = Dog ("Doug", 4)
william = Dog ("William", 5)

# Determine the oldest dog
def get_biggest_number (* args):
    return max (args)

# Output
print ("The oldest dog is {} years old.".format (
    get_biggest_number (jake.age, doug.age, william.age)))

```

Metodi di istanza

I metodi di istanza sono definiti all'interno della classe e vengono utilizzati per ottenere il contenuto dell'istanza. Possono anche essere utilizzati per eseguire operazioni con l'attributo dei nostri oggetti. Come un metodo init, il primo argomento è sempre self:

```

class Dog:

    # Class Attribute
    species = 'mammal'

    # Initializer / Instance Attributes
    def __init__(self, name, age):
        self.name = name
        self.age = age

    # instance method
    def description (self):
        return " {} is {} years old" .format (self.name, self.age)

    # instance method
    def speak (self, sound):
        return " {} says {}".format (self.name, sound)

# Instantiate the Dog object
mikey = Dog ("Mikey", 6)

# call our instance methods
print (mikey.description ())
print (mikey.speak ("Gruff Gruff"))

```

Salva come dog_instance_methods.py , quindi esegilo:

```

Mikey is 6 years old
Mikey says Gruff Gruff

```

Nell'ultimo metodo, speak() definiamo il comportamento. Quali altri tipi di

comportamento si può assegnare a un cane? Torna all'inizio del paragrafo per vedere esempi del comportamento di altri oggetti.

Modifica attributo

Puoi cambiare il valore degli attributi in base ad alcuni comportamenti: >>>

```
>>> class Email:
...     def __init__(self):
...         self.is_sent = False
...     def send_email(self):
...         self.is_sent = True
...
>>> my_email = Email()
>>> my_email.is_sent
False
>>> my_email.send_email()
>>> my_email.is_sent
True
```

Qui abbiamo aggiunto un metodo per inviare un'e-mail che aggiorna `is_sent` variable su `True`.

Ereditarietà degli oggetti Python

L'ereditarietà è un'elaborazione in cui una classe accetta gli attributi e i metodi di un'altra. Le classi appena create sono chiamate classi figlio e le classi da cui derivano le classi figlio sono chiamate classi padre.

È importante notare che le classi figlie sovrascrivono o estendono la funzionalità (ad esempio, attributi e comportamento) delle classi padre. In altre parole, le classi figlie ereditano tutti gli attributi e il comportamento del genitore, ma possono anche definire altri comportamenti da seguire. Il tipo più elementare di classe è l'oggetto classe, che di solito tutte le altre classi ereditano come genitori.

Quando definisci una nuova classe, Python 3 usa implicitamente il suo oggetto come classe genitore. Pertanto, le seguenti due definizioni sono equivalenti:

```
class Dog(object):
    pass

# In Python 3, this is the same as:

class Dog:
    pass
```

Nota. In Python 2.x, c'è una differenza tra [le classi nuove e vecchie](#). Non entrerà nei dettagli, ma di solito vorrai specificare un oggetto come classe genitore per assicurarti di definire una nuova classe di stile se stai scrivendo codice Python 2 OOP.

Esempio di parco per cani

Immaginiamo di essere in un parco per cani. Ci sono diversi oggetti Cane coinvolti nel comportamento del Cane, ognuno con attributi diversi. Nella normale conversazione, questo significa che alcuni cani corrono, altri sono allungati e alcuni stanno solo guardando altri cani. Inoltre, ogni cane è stato chiamato il suo proprietario, e poiché ogni cane vive e respira, ognuno sta invecchiando.

In quale altro modo puoi distinguere un cane da un altro? Che ne dici di una razza di cane: >>>

```
>>> class Dog:
...     def __init__(self, breed):
...         self.breed = breed
...
>>> spencer = Dog("German Shepard")
>>> spencer.breed
'German Shepard'
>>> sara = Dog("Boston Terrier")
>>> sara.breed
'Boston Terrier'
```

Ogni razza di cane ha comportamenti leggermente diversi. Per tenerne conto, creiamo classi separate per ogni razza. Queste sono le classi figlie della Dogclass genitore.

Estensione della funzionalità della classe genitore

Crea un nuovo file chiamato dog_inheritance.py :

```

# Parent class
class Dog:

    # Class attribute
    species = 'mammal'

    # Initializer / Instance attributes
    def __init__(self, name, age):
        self.name = name
        self.age = age

    # instance method
    def description (self):
        return "{} is {} years old".format(self.name, self.age)

    # instance method
    def speak (self, sound):
        return "{} says {}".format(self.name, sound)

# Child class (inherits from Dog class)
class RussellTerrier (Dog):
    def run (self, speed):
        return "{} runs {}".format(self.name, speed)

# Child class (inherits from Dog class)
class Bulldog (Dog):
    def run (self, speed):
        return "{} runs {}".format(self.name, speed)

# Child classes inherit attributes and
# behaviors from the parent class
jim = Bulldog ("Jim", 12)
print (jim.description ())

# Child classes have specific attributes
# and behaviors as well
print (jim.run ("slowly"))

```

Leggi i commenti ad alta voce mentre lavori con questo programma per aiutarti a capire cosa sta succedendo, quindi, prima di eseguire il programma, verifica se puoi prevedere il risultato previsto.

Tu dovresti vedere:

```

Jim is 12 years old
Jim runs slowly

```

Non abbiamo aggiunto attributi o metodi speciali per distinguere tra un RussellTerrier e un Bulldog. Tuttavia, poiché ora sono due classi diverse, potremmo, ad esempio, dare loro attributi di classe diversi che determinano le loro rispettive velocità.

Classi genitori e figli

`isinstance()` La funzione viene utilizzata per determinare se l'istanza è anche un'istanza di una specifica classe genitore. Salva questo come `dog_isinstance.py` :

```

# Parent class
class Dog:

    # Class attribute
    species = 'mammal'

    # Initializer / Instance attributes
    def __init__(self, name, age):
        self.name = name
        self.age = age

    # instance method
    def description (self):
        return "{} is {} years old".format (self.name, self.age)

    # instance method
    def speak (self, sound):
        return "{} says {}".format (self.name, sound)

# Child class (inherits from Dog () class)
class RussellTerrier (Dog):
    def run (self, speed):
        return "{} runs {}".format (self.name, speed)

# Child class (inherits from Dog () class)
class Bulldog (Dog):
    def run (self, speed):
        return "{} runs {}".format (self.name, speed)

# Child classes inherit attributes and
# behaviors from the parent class
jim = Bulldog ("Jim", 12)
print (jim.description ())

# Child classes have specific attributes
# and behaviors as well
print (jim.run ("slowly"))

# Is jim an instance of Dog ()?
print (isinstance (jim, Dog))

# Is julie an instance of Dog ()?
julie = Dog ("Julie", 100)
print (isinstance (julie, Dog))

# Is johnny walker an instance of Bulldog ()
johnnywalker = RussellTerrier ("Johnny Walker", 4)
print (isinstance (johnnywalker, Bulldog))

# Is julie and instance of jim?
print (isinstance (julie, jim))

```

Conclusion: >>>


```
('Jim', 12)
Jim runs slowly
True
True
False
Traceback (most recent call last):
  File "dog_isinstance.py", line 50, in <module>
    print (isinstance (julie, jim))
TypeError: isinstance () arg 2 must be a class, type, or tuple of classes and types
```

Ha senso? Sia jim and julie sono istanze della classe Dog() che johnnywalker non sono istanze della classe Bulldog(). Quindi, come controllo dello stato, abbiamo verificato julie se l'istanza è un'istanza jim, il che è impossibile, poiché l'istanza jimit non appartiene alla classe stessa, ma alla classe TypeError.

Ignorare la funzionalità della classe padre

Ricorda che le classi figlie possono anche sovrascrivere gli attributi e il comportamento della classe padre. Ad esempio: >>>

```
>>> class Dog:
... species = 'mammal'
...
>>> class SomeBreed (Dog):
... pass
...
>>> class SomeOtherBreed (Dog):
... species = 'reptile'
...
>>> frank = SomeBreed ()
>>> frank.species 'mammal'
>>> beans = SomeOtherBreed ()
>>> beans.species 'reptile'
The SomeBreed()class inherits speciesfrom the parent class,
while the SomeOtherBreed()class overrides speciesby setting it
reptile.
```

Revisione degli esercizi (n. 2)

Esercizio: "Eredità dei cani"

Crea una Petsclass che contenga istanze di cani: questa classe è completamente

separato dalla Dogclass. In altre parole, una Dogclass non viene ereditata dalla Petsclass. Quindi assegna tre istanze del cane all'istanza Petsclass . Inizia con il seguente codice qui sotto. Salva il file come Pets_class.py . Il tuo output dovrebbe essere simile a questo:

```
I have 3 dogs.  
Tom is 6.  
Fletcher is 7.  
Larry is 9.  
And they're all mammals, of course.
```

Codice di inizio:

```
# Parent class  
class Dog:  
  
    # Class attribute  
    species = 'mammal'  
  
    # Initializer / Instance attributes  
    def __init__(self, name, age):  
        self.name = name  
        self.age = age  
  
    # instance method  
    def description(self):  
        return "{} is {} years old".format(self.name, self.age)  
  
    # instance method  
    def speak(self, sound):  
        return "{} says {}".format(self.name, sound)  
  
# Child class (inherits from Dog class)  
class RussellTerrier(Dog):  
    def run(self, speed):  
        return "{} runs {}".format(self.name, speed)  
  
# Child class (inherits from Dog class)  
class Bulldog(Dog):  
    def run(self, speed):  
        return "{} runs {}".format(self.name, speed)
```

Soluzione: "Eredità del cane"

```
# Parent class
class Pets:

    dogs = []

    def __init__(self, dogs):
        self.dogs = dogs


# Parent class
class Dog:

    # Class attribute
    species = 'mammal'

    # Initializer / Instance attributes
    def __init__(self, name, age):
        self.name = name
        self.age = age

    # Instance method
    def description(self):
        return self.name, self.age

    # Instance method
    def speak(self, sound):
        return "% s says% s" % (self.name, sound)
```

```

# Instance method
def eat (self):
    self.is_hungry = False

# Child class (inherits from Dog class)
class RussellTerrier (Dog):
    def run (self, speed):
        return "% s runs% s" % (self.name, speed)

# Child class (inherits from Dog class)
class Bulldog (Dog):
    def run (self, speed):
        return "% s runs% s" % (self.name, speed)

# Create instances of dogs
my_dogs = [
    Bulldog ("Tom", 6),
    RussellTerrier ("Fletcher", 7),
    Dog ("Larry", 9)
]

# Instantiate the Pets class
my_pets = Pets (my_dogs)

# Output
print ("I have {} dogs.".format (len (my_pets.dogs)))
for dog in my_pets.dogs:
    print (" {} is {}".format (dog.name, dog.age))

print ("And they're all {} s, of course.".format (dog.species))

```

Esercizio: cani affamati

Usa gli stessi file, aggiungi un attributo di istanza `is_hungry = True` nella Dogclass. Quindi aggiungi il metodo chiamato, `eat()` che quando viene chiamato cambia il valore `is_hungry` to `False`. Scopri come nutrire al meglio ogni cane, quindi stampa "I miei cani hanno fame". se tutti hanno fame o "I miei cani non hanno fame". se non tutti hanno fame. Il risultato finale dovrebbe assomigliare a questo:

```

I have 3 dogs.
Tom is 6.
Fletcher is 7.
Larry is 9.
And they're all mammals, of course.
My dogs are not hungry.

```

Soluzione: cani affamati

```
# Parent class
```

```
class Pets:
```

```
    dogs = []
```

```
    def __init__(self, dogs):  
        self.dogs = dogs
```

```
# Parent class
```

```
class Dog:
```

```
    * Class attribute
```

```
    species = 'mammal'
```

```
    * Initializer." In instance attributes
```

```
    def __init__(self, name, age):  
        self.name = name  
        self.age = age  
        self.is_hungry = True
```

```
= instance method
```

```
    def description(self):  
        return self.name, self.age
```

```
= instance method
```

```
    def speak(self, sound):  
        return "%s says '%s'." % (self.name, sound)
```

```

= mist ane method
def eat (self).
    self.is_hungry = False

```

```

= Oid class (inherits from Dog class)
class RussellTerrier (Dog).
    def run (self speed).
        return " %s runs %s" % (self.name, speed)

```

```

= Oid class (inherits from Dog class)
class Bulldog (Dog).
    def run (self speed).
        return " %s runs %s" % (self.name, speed)

```

```

= Create instances of dogs
my_dogs = [
    Bulldog ("Tom" 6)
    RussellTerrier ("Fletcher" 10)
    Dog ("Lare" 9)

```

```

# instantiate the Pets class
myPets = Pets (my_dogs)

```

```

= Output
print (" I have { } dogs.".format (len (myPets.dogs)))
for dog in myPets.dogs:
    dog.eat ()
    print (" { } is { }.".format (dog.name, dog.age))

print ("And the { } of { } exercise.".format (dog.species))

```

```

are my_dogs hungry? = False
for dog in myPets.dogs:
    if dog.is_hungry:
        are_my_dogs_hungry = True

if are_my_dogs_hungry:
    print ("My dogs are hungry.")
else:
    print ("My dogs are not hungry.")

```

Esercizio: "Passeggiata con il cane"

Successivamente, aggiungi un metodo walk() come Petse Dogclasses, in modo che quando chiami un metodo nella Petsclass, ogni istanza di un cane venga assegnata alla classe Petsa walk(). Salva questo come dog_walking.py . Questo è un po' più complicato.

Inizia implementando un metodo proprio come un metodo speak(). Per quanto riguarda la

metodo nella Petsclass, dovrai scorrere l'elenco dei cani e quindi chiamare il metodo stesso.

L'output dovrebbe essere simile a questo:

```
Tom is walking!  
Fletcher is walking!  
Larry is walking!
```

Soluzione: "dog walking"

```
# Parent class  
class Pets:  
  
    dogs = []  
  
    def __init__(self, dogs):  
        self.dogs = dogs  
  
    def walk(self):  
        for dog in self.dogs:  
            print(dog.walk())  
  
  
# Parent class  
class Dog:  
  
    # Class attribute  
    species = 'mammal'  
    is_hungry = True  
  
    # Initializer / instance attributes  
    def __init__(self, name, age):  
        self.name = name  
        self.age = age  
  
    # Instance method  
    def description(self):  
        return self.name, self.age  
  
    # Instance method  
    def speak(self, sound):  
        return "%s says %s" % (self.name, sound)  
  
    # Instance method  
    def eat(self):  
        self.is_hungry = False
```

```

def walk (self):
    return "% s is walking!" % (self.name)

# Child class (inherits from Dog class)
class RussellTerrier (Dog):
    def run (self, speed):
        return "% s runs% s" % (self.name, speed)

# Child class (inherits from Dog class)
class Bulldog (Dog):
    def run (self, speed):
        return "% s runs% s" % (self.name, speed)

# Create instances of dogs
my_dogs = [
    Bulldog ("Tom", 6),
    RussellTerrier ("Fletcher", 7),
    Dog ("Larry", 9)
]

# Instantiate the Pet class
my_pets = Pets (my_dogs)

# Output
my_pets.walk ()

```

Esercizio: "Verificare la comprensione"

Rispondi alle seguenti domande OOP per verificare i tuoi progressi di apprendimento:

1. Quale classe?
2. Che esempio?
3. Qual è la relazione tra classe e istanza?
4. Quale sintassi Python viene utilizzata per definire una nuova classe?
5. Qual è la convenzione ortografica per il nome di una classe?
6. Come si crea o si crea un'istanza di una classe?
7. Come si accede agli attributi e al comportamento di un'istanza di una classe?
8. Che tipo di metodo?
9. Qual è lo scopo del sé?
10. Qual è lo scopo del metodo init?
11. Descrivi come l'ereditarietà aiuta a prevenire il codice duplicazione.

12. Le classi figlie possono sovrascrivere le proprietà delle loro? genitori?

Soluzione: "Test di comprensione" Mostra nascondi

1. Un meccanismo di classe utilizzato per creare nuove strutture dati personalizzate. Contiene dati, nonché i metodi utilizzati per elaborare questi dati.
2. Un'istanza è una copia di una classe con valori effettivi, letteralmente un oggetto di una particolare classe.
3. Mentre una classe è un piano utilizzato per descrivere come creare qualcosa, le istanze sono oggetti creati da questi disegni.
4. `class PythonClassName:`
5. Designazione CamelCase in maiuscolo, ad esempio `PythonClassName()`
6. Si utilizza il nome della classe seguito da parentesi. Quindi se il nome della classe `Dog()`, l'istanza del cane sarà - `mia_classe = Cane()`.
7. Con punto `nome_istanza.nome_attributo` notazione - per esempio,
8. Una funzione definita all'interno di una classe.
9. Il primo argomento di ogni metodo si riferisce all'istanza corrente della classe, che viene chiamata dalla convenzione `self`. Il metodo `init` `self` fa riferimento a un oggetto appena creato, mentre in altri metodi fa riferimento all'istanza il cui metodo è stato chiamato. Per ulteriori informazioni su `init` con `self`, dai un'occhiata [questo](#) articolo.
10. `init` Il metodo inizializza un'istanza della classe.
11. Le classi figlie ereditano tutti gli attributi e il comportamento di il genitore.
12. Sì.

CONCLUSIONE

Ora dovresti sapere cosa sono le classi, perché vuoi o dovresti usarle e come creare classi genitore e figlio per strutturare meglio i tuoi programmi.

Ricorda che OOP è un paradigma di programmazione, non un concetto Python. La maggior parte dei linguaggi di programmazione moderni, come Java, C#, C++, seguono i principi dell'OOP. Quindi la buona notizia è che imparare le basi della programmazione orientata agli oggetti ti sarà utile in una varietà di circostanze, indipendentemente dal fatto che lavori in Python o meno.

Ora il settore non si ferma e sono sempre di più i siti web, i servizi e le aziende che hanno bisogno di specialisti.

La domanda di sviluppatori sta crescendo, ma la concorrenza tra di loro sta crescendo.

Per essere il migliore nella tua attività, devi essere una persona quasi assolutamente universale in grado di scrivere un sito Web, creare un design per esso e promuoverlo da solo.

A tal proposito anche una persona che non si è mai seduta davanti a questo computer comincia a pensare, ma devo provare?

Ma molto spesso si scopre che tali entusiasti si esauriscono nella fase iniziale, senza essersi cimentati in questa materia.

O forse sarebbe diventato un geniale creatore di codice? Creerebbe qualcosa di nuovo? Questo non lo sapremo.

Ogni giorno cresce la soglia per entrare in programmazione. Non puoi mai prevedere quale nuova lingua uscirà.

Una tale abbondanza rompe tutto il desiderio di un programmatore appena coniato e si perde in questo oceano di informazioni.

Tutti questi tuoi javascript... pitoni... che paura..

Un grande equivoco è l'obbligo di conoscere la matematica. Sì, serve, ma solo in un campo ristretto, ma è anche utile per capire alcuni aspetti.

Il consiglio che si può dare alle persone che stanno appena iniziando la loro attività è

per non inseguire tutto in una volta. Concediti del tempo per pensare.

Cosa voglio fare? Creare un programma per essere utile a tutti? Creare servizi aggiuntivi per semplificare qualsiasi attività? O devi davvero andare a fare giochi?

Il secondo consiglio sarà quello di rispondere alla mia domanda quanto tempo sono pronto a dedicare a questo? Il terzo punto è pensare a quanto velocemente vuoi ottenere il risultato desiderato.

Quindi, non esiste un "modo semplice" per iniziare a programmare, tutto dipende da te: dal tuo interesse, da cosa vuoi fare e dai tuoi compiti.

In ogni caso, devi cercare di fare tutto il possibile in tuo potere. Buona fortuna per il tuo impegno!

PYTHON PER LA SCIENZA DEI DATI

Guida alla programmazione del computer e alla codifica web. Impara l'apprendimento automatico, l'intelligenza artificiale, i pacchetti NumPy e Pandas per l'analisi dei dati. Esercizi passo passo incluso.

JASON TEST

INTRODUZIONE

Data Science è stato molto popolare negli ultimi due anni. L'obiettivo principale di questo settore è incorporare dati significativi nelle strategie commerciali e di marketing che aiuteranno l'azienda a espandersi. E arrivare a una soluzione logica, i dati possono essere archiviati ed esplorati.

Originariamente solo le principali società IT erano impegnate in questo campo, ma oggi la tecnologia dell'informazione viene utilizzata da aziende che operano in diversi settori e campi come l'e-commerce, l'assistenza medica, i servizi finanziari e altri. Sono disponibili programmi di elaborazione software come Hadoop, codice R, SAS, SQL e molti altri. Python è, tuttavia, lo strumento di dati e analisi più famoso e più facile da usare. È riconosciuto come il coltellino svizzero del mondo della codifica poiché promuove la codifica strutturata, la programmazione orientata agli oggetti, il linguaggio di programmazione operativa, e molti altri. Python è il linguaggio di programmazione più utilizzato al mondo ed è anche riconosciuto come il linguaggio di più alto livello per gli strumenti e le tecniche di data science, secondo lo studio Stack Overflow del 2018.

Nel sondaggio per sviluppatori Hacker rank 2018, che si vede nella loro classifica amore-odio, Python ha conquistato il cuore dello sviluppatore. Gli esperti di data science si aspettano di vedere un aumento dell'ecosistema Python, con una crescente popolarità. E sebbene il tuo viaggio nello studio della programmazione Python possa essere appena iniziato, è bello sapere che ci sono anche molte (e in aumento) opzioni di carriera.

Analisi dei dati La programmazione Python è ampiamente utilizzata e, oltre ad essere un linguaggio flessibile e open source, diventa uno dei linguaggi di programmazione preferiti. Le sue grandi librerie vengono utilizzate per l'elaborazione dei dati e, anche per un analista di dati principiante, sono molto facili da capire. Oltre ad essere open-source, si integra facilmente anche con qualsiasi infrastruttura che possa essere utilizzata per risolvere i problemi più complicati. Viene utilizzato dalla maggior parte delle banche per l'elaborazione dei dati, dalle organizzazioni per l'analisi e l'elaborazione e le società di previsione meteorologica come l'analisi dei monitor climatici spesso lo utilizzano. Il salario annuale per un informatico è di \$ 127.918, secondo Indeed. Quindi ecco la buona notizia, la cifra è destinata ad aumentare. Gli esperti di IBM prevedono un aumento del 28% delle richieste dei data scientist entro il 2020. Per la data science, tuttavia, il futuro è luminoso, e Python è solo una fetta della torta dorata. Fortunatamente padroneggiare Python e altri principi di programmazione sono pratici come non mai.

LA SCIENZA DEI DATI E IL SUO SIGNIFICATO

La Data Science ha fatto molta strada negli ultimi anni e, quindi, diventa un fattore importante per comprendere il funzionamento di più aziende. Di seguito sono riportate diverse spiegazioni che dimostrano che la scienza dei dati sarà ancora parte integrante del mercato globale.

1. Le aziende sarebbero in grado di comprendere il loro cliente in modo più efficiente ed elevato con l'aiuto della Data Science. I clienti soddisfatti costituiscono la base di ogni azienda e svolgono un ruolo importante nei loro successi o insuccessi. Data Science consente alle aziende di interagire con i clienti in modo anticipato e quindi dimostra le prestazioni e la forza migliorate del prodotto.

2. La Data Science consente ai marchi di offrire immagini potenti e accattivanti. Questo è uno dei motivi è famoso. Quando i prodotti e le aziende fanno un uso inclusivo di questi dati, possono condividere le loro esperienze con il loro pubblico e creare così relazioni migliori con l'articolo.

3. Forse una delle caratteristiche significative di Data Science è che i suoi risultati possono essere generalizzati a quasi tutti i tipi di industrie, come i viaggi, l'assistenza sanitaria e l'istruzione. Le aziende possono determinare rapidamente i loro problemi con l'aiuto della Data Science e possono anche affrontarli adeguatamente

4. Attualmente, la scienza dei dati è accessibile in quasi tutti i settori e al giorno d'oggi ce n'è una quantità enorme di dati esistenti nel mondo e, se utilizzato adeguatamente, può portare alla vittoria o al fallimento di qualsiasi progetto. Se i dati vengono utilizzati correttamente, sarà importante in futuro raggiungere gli obiettivi del prodotto.

5. I big data sono sempre in aumento e in crescita. I big data consentono all'azienda di affrontare problemi Problemi di business, capitale umano e gestione del capitale in modo efficace e rapido utilizzando diverse risorse che vengono costruite di routine.

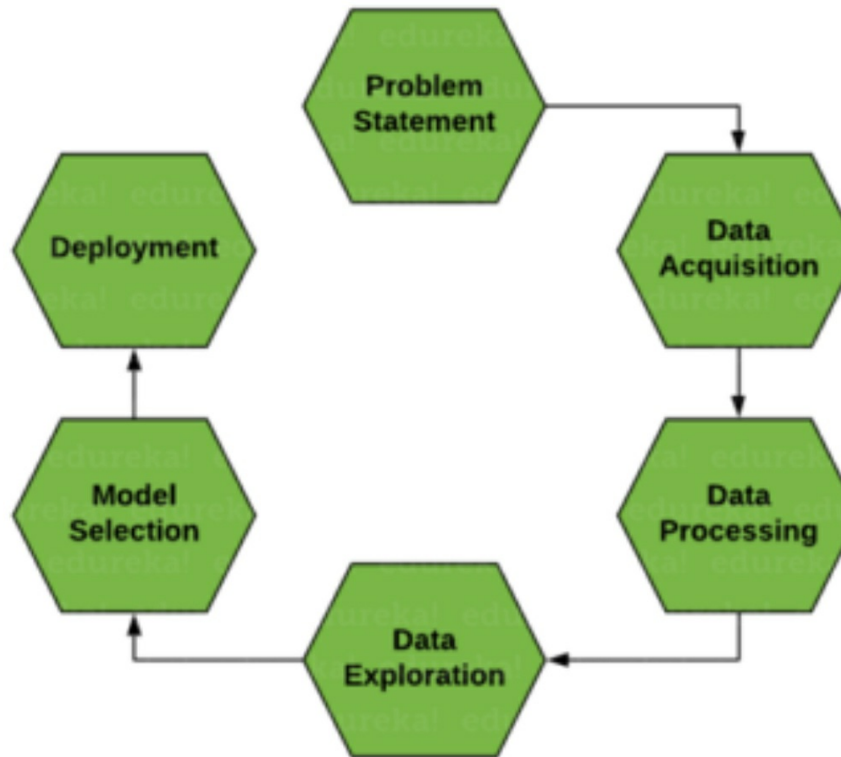
6. La scienza dei dati sta guadagnando rapidamente popolarità in ogni altro settore e svolge quindi un ruolo importante nel funzionamento e nelle prestazioni di ogni prodotto. Pertanto, anche il ruolo del data scientist viene rafforzato poiché svolgerà una funzione essenziale di gestione dei dati e di fornire soluzioni a problemi particolari.

7. La tecnologia informatica ha colpito anche i settori dei supermercati. Per capirlo, prendiamo un esempio le persone anziane hanno avuto una fantastica interazione con il venditore locale. Inoltre, il venditore è stato in grado di soddisfare le esigenze dei clienti in modo personalizzato. Ma ora questa attenzione è andata persa a causa della nascita e dell'aumento delle catene di supermercati. Ma i venditori sono in grado di comunicare con i propri clienti con l'aiuto dell'analisi dei dati.

8. Data Science aiuta le aziende a creare quella connessione con i clienti. Le aziende e le loro merci saranno in grado di avere una comprensione migliore e più profonda di come i clienti possono utilizzare i loro servizi con l'aiuto della scienza dei dati.

Futuro della tecnologia dei dati: come altre aree in continua evoluzione, anche l'importanza della tecnologia dei dati sta crescendo sempre di più. La scienza dei dati ha avuto un impatto su diversi campi. La sua influenza può essere vista in molti settori, come la vendita al dettaglio, la sanità e l'istruzione. Nuovi trattamenti e tecnologie vengono continuamente identificati nel settore sanitario ed è necessaria un'assistenza di qualità ai pazienti. Il settore sanitario può trovare una soluzione con l'aiuto di tecniche di data science che aiutano i pazienti a prendersi cura di loro. L'istruzione è un altro campo in cui si possono vedere chiaramente i vantaggi della scienza dei dati. Ora le nuove innovazioni come telefoni e tablet sono diventate una caratteristica essenziale del sistema educativo. Inoltre, con l'aiuto della scienza dei dati, gli studenti stanno creando maggiori possibilità, il che porta a migliorare le loro conoscenze.

Ciclo di vita della scienza dei dati:



Strutture dati

Una struttura di dati può essere selezionata nella programmazione del computer o progettata per memorizzare dati allo scopo di lavorare con diversi algoritmi su di essa. Ogni altra struttura dati include i valori dei dati, le relazioni dei dati e le funzioni tra i dati che possono essere applicati ai dati e alle informazioni.

Caratteristiche delle strutture dati

A volte, le strutture dati sono classificate in base alle loro caratteristiche. Le funzioni possibili sono:

- Lineare o non lineare: questa funzionalità definisce come gli oggetti dati sono organizzati in una serie sequenziale, come un elenco o in una sequenza non ordinata, come una tabella.
- Omogeneo o non omogeneo: questa funzione definisce come tutti gli oggetti dati in una raccolta sono dello stesso tipo o di tipo diverso.
- Statico o dinamico: questa tecnica determina di mostrare di assemblare le strutture dati. Le strutture dati statiche al momento della compilazione hanno dimensioni, strutture e destinazioni fisse nella memoria. I tipi di dati dinamici hanno dimensioni, meccanismi e destinazioni di memoria che possono ridursi o espandersi a seconda dell'applicazione.

Tipi di strutture dati

I tipi della struttura dati sono determinati da quali tipi di operazioni saranno necessarie o da quali tipi di algoritmi verranno implementati. Ciò comprende:

Array: un array memorizza un elenco di elementi di memoria in posizioni adiacenti. I componenti della stessa categoria si trovano insieme poiché la posizione di ciascun elemento può essere facilmente calcolata o accessibile. matrici

può essere di dimensioni fisse o flessibile in lunghezza.

Pile: una pila contiene un insieme di oggetti in ordine lineare aggiunti alle operazioni. Questo ordine può essere scaduto in first out (LIFO) o first-out (FIFO).

Code: una coda memorizza una selezione di elementi simile a uno stack; tuttavia, la sequenza di attività può essere solo first in first out. **Elenchi collegati:** in un ordine lineare, un elenco collegato memorizza una selezione di elementi. In un elenco collegato, ogni unità o nodo include un elemento di dati nonché un riferimento o una relazione all'elemento successivo nell'elenco.

Alberi: un albero contiene una raccolta astratta e gerarchica di elementi. Ogni nodo è connesso ad altri nodi e può avere diversi sottovalori, detti anche figlio.

Grafici: un grafico memorizza un gruppo di elementi di progettazione non lineare. I grafici sono costituiti da un insieme limitato di nodi, chiamati anche vertici, e da linee che li collegano, note anche come bordi. Sono utili per descrivere i processi nella vita reale, come i computer in rete.

Tentativi: un tria o un albero di query è spesso una struttura di dati che memorizza le stringhe come file di dati, che possono essere organizzati in un grafico visivo.

Tabelle hash: una tabella hash o un grafico hash è contenuto in un elenco relazionale che etichetta le chiavi delle variabili. Una tabella hash utilizza un algoritmo di hashing per trasformare un indice in una matrice di contenitori contenenti l'elemento di dati desiderato. Questi sistemi di dati sono chiamati complessi perché possono contenere grandi quantità di dati interconnessi. Esempi di strutture dati primali o fondamentali sono interi, float, booleani e caratteri.

Utilizzo di strutture dati

Le strutture di dati sono generalmente utilizzate per incorporare i tipi di dati in forme fisiche. Questo può essere interpretato in un'ampia gamma di applicazioni, incluso un albero binario che mostra una tabella di database. Le strutture dati sono utilizzate nei linguaggi di programmazione per organizzare codice e informazioni nella memoria digitale. I database e i dizionari Python, o array e oggetti JavaScript, sono sistemi di codifica popolari utilizzati per raccogliere e analizzare i dati. Inoltre, le strutture dati sono una parte vitale di una progettazione software efficace. Importanza dei database I sistemi di dati sono necessari per gestire in modo efficace vasti volumi di dati, come i dati archiviati nelle biblioteche o i servizi di indicizzazione.

La gestione accurata della configurazione dei dati richiede identificatori di allocazione della memoria, interconnessioni di dati e processi di dati, che supportano tutte le strutture di dati. Inoltre, è importante non solo utilizzare le strutture dati, ma anche selezionare la struttura dati corretta per ogni assegnazione.

La scelta di una struttura dati insoddisfacente potrebbe comportare tempi di esecuzione lenti o codice disorientato. Tutte le considerazioni che devono essere prese in considerazione quando si sceglie un sistema di dati includono il tipo di informazioni che devono essere elaborate, dove verranno inseriti i nuovi dati, come verranno organizzati i dati e quanto spazio verrà assegnato ai dati.

BASI DI PITONE

Sì Puoi ottenere tutte le conoscenze sul linguaggio di programmazione Python in 5 semplici passaggi.

Passaggio 1: esercitazioni di base in Python

Tutto inizia da qualche parte. Questo primo passo è dove verranno apprese le basi della programmazione di Python. Avrai sempre bisogno di un'introduzione alla scienza dei dati. Jupyter Notebook, che viene fornito con le librerie Python per aiutarti a comprendere questi due fattori, che lo rendono una delle risorse essenziali che puoi iniziare a utilizzare all'inizio del tuo viaggio.

Passaggio 2: prova a fare pratica con i progetti Mini-Python

Crediamo fortemente nell'apprendimento attraverso le spalle. Prova a programmare cose come giochi su Internet, calcolatrici o software che ottengono il meteo di Google nella tua zona. La creazione di questi mini-progetti può aiutarti a capire Python. Progetti come questi sono standard per qualsiasi linguaggio di programmazione e un modo fantastico per rafforzare le tue conoscenze lavorative. Avrai una conoscenza API migliore e avanzata e continuerai a raschiare il sito con tecniche avanzate. Ciò ti consentirà di apprendere la programmazione Python in modo più efficace e il metodo di web scraping ti sarà utile in seguito durante la raccolta dei dati.

Fase 3: impara le biblioteche scientifiche su Python

Python può fare qualsiasi cosa con i dati. Pandas, Matplotlib e NumPy sono noti per essere le tre librerie Python più utilizzate e più importanti per la scienza dei dati. NumPy e Pandas sono utili per la creazione e lo sviluppo dei dati. Matplotlib è una libreria per analizzare i dati, creare diagrammi di flusso e diagrammi come vorresti vedere in Excel o Fogli Google.

Fase 4: creare un portfolio

Un portfolio è una necessità assoluta per i data scientist professionisti. Questi progetti devono includere numerosi set di dati e lasciare ai lettori importanti prospettive che hai raccolto. Il tuo portfolio non ha un tema specifico; trovare set di dati che ti ispirano e quindi trovare un modo per metterli insieme. Mostrare progetti come questi fornisce una certa collaborazione ai colleghi scienziati dei dati e dimostra ai futuri datori di lavoro che hai davvero colto l'occasione per comprendere Python e altre competenze di codifica essenziali. Alcuni degli aspetti positivi della scienza dei dati sono che, mentre mostri le abilità che hai appreso, il tuo portfolio funge da curriculum, come la programmazione in Python.

Passaggio 5: applicare tecniche avanzate di data science

Alla fine, l'obiettivo è rafforzare le tue capacità di programmazione. Il tuo percorso di data science sarà pieno di apprendimento continuo, ma puoi realizzare tutorial specializzati per assicurarti di esserti specializzato nella programmazione di base di Python. Devi prendere confidenza con i modelli di clustering di regressione, raggruppamento e k-mean. Puoi anche tuffarti nell'apprendimento automatico utilizzando lezioni di sci-kit per avviare i modelli e creare modelli di rete neurale. A questo punto, la programmazione per sviluppatori potrebbe includere la creazione di modelli utilizzando origini dati in tempo reale. Questo tipo di tecnica di apprendimento automatico modifica le sue ipotesi nel tempo.

Quanto è importante Python per la scienza dei dati?

Efficiente e semplice da usare: Python è considerato uno strumento per i principianti e per qualsiasi studente o

il ricercatore con solo una conoscenza di base potrebbe iniziare a lavorarci. Anche il tempo e il denaro spesi per il debug di codici e vincoli sulla gestione di progetti diversi sono ridotti al minimo. Il tempo per l'implementazione del codice è inferiore rispetto ad altri linguaggi di programmazione come C, Java e C#, il che fa sì che gli sviluppatori e gli ingegneri del software trascorrono molto più tempo a lavorare sui loro algoritmi.

Library Choice-Python offre una vasta libreria e machine learning e intelligenza artificiale Banca dati. Scikit Learn, TensorFlow, Seaborn, Pytorch, Matplotlib e molti altri sono tra le librerie più popolari. Ci sono molti video tutorial online e risorse sull'apprendimento automatico e sulla scienza dei dati, che possono essere facilmente ottenuti.

Scalabilità: Python ha dimostrato di essere un linguaggio altamente scalabile e più veloce rispetto ad altri linguaggi di programmazione come c++, Java e R. Offre flessibilità nella risoluzione di problemi che non possono essere risolti con altri linguaggi per computer. Molte aziende lo utilizzano per sviluppare tutti i tipi di tecniche e sistemi rapidi.

Visual Statistics and Graphics-Python fornisce una serie di strumenti di visualizzazione. Il Matplotlib library fornisce un framework affidabile su cui vengono sviluppate quelle librerie come gg plot, pandas plotting, PyTorch e altri. Questi servizi aiutano a creare grafici, linee di trama pronte per il Web, layout visivi, ecc.

Come viene utilizzato Python per la scienza dei dati

Prima fase – Prima di tutto, dobbiamo imparare e capire che forma prende un dato. Se percepiamo dati per essere un enorme foglio Excel con colonne e corvi lakh, quindi forse dovresti sapere cosa fare al riguardo? È necessario raccogliere informazioni in ogni riga e colonna eseguendo alcune operazioni e cercando un tipo specifico di dati. Il completamento di questo tipo di attività computazionale può richiedere molto tempo e duro lavoro. Pertanto, puoi utilizzare le librerie di Python, come Pandas e Numpy, che possono completare rapidamente le attività utilizzando il calcolo parallelo.

Seconda fase – Il prossimo ostacolo è ottenere i dati necessari. Poiché i dati non sono sempre disponibili accessibile a noi, dobbiamo scaricare i dati dalla rete secondo necessità. Qui Python Scrap e le brillanti librerie Soup possono permetterci di recuperare dati da Internet.

Terza fase – In questa fase dobbiamo ottenere la simulazione o la presentazione visiva dei dati. Guida prospettive diventa difficile quando hai troppe figure sul tabellone. Il modo corretto per farlo è rappresentare i dati in forma di grafico, grafici e altri layout. Le librerie Python Seaborn e Matplotlib vengono utilizzate per eseguire questa operazione.

Quarta fase: la fase successiva è l'apprendimento automatico, che è un'elaborazione estremamente complicata. Esso include strumenti matematici come le operazioni di probabilità, calcolo e matrice di colonne e righe su lakh. Con la libreria di apprendimento automatico di Python Scikit-Learn, tutto questo diventerà molto semplice ed efficace.

Libreria standard

La libreria Python Standard è costituita dalla sintassi, dal token e dalla semantica precisi di Python. Viene fornito con il core di distribuzione Python. Quando abbiamo iniziato con un'introduzione, abbiamo fatto riferimento a questo. È scritto in C e copre funzionalità come I/O e altri componenti principali. Insieme, tutta la versatilità del rendering rende Python il linguaggio che è. Alla radice della libreria di base, ci sono più di 200 moduli chiave. Python fornisce quella libreria. Ma oltre a questa libreria, puoi anche ottenere un'enorme raccolta di diverse migliaia di componenti Python Package Index (PyPI).

1. Matplotlib

'Matplotlib' aiuta ad analizzare i dati ed è una libreria di grafici numerici. Per Data Science, abbiamo discusso in Python.

2. Panda

'Panda' è un must per la scienza dei dati, come abbiamo detto prima. Fornisce strutture di dati facili, descrittive e versatili per gestire con facilità (e fluidamente) dati organizzati (tabulati, multistrato, presumibilmente eterogenei) e in serie.

3. Richieste

"Requests" è una libreria Python che consente agli utenti di caricare richieste HTTP/1.1, aggiungere intestazioni, dati del modulo, file multipart e semplici parametri del dizionario Python. Allo stesso modo, ti aiuta anche ad accedere ai dati di risposta.

4. NumPy

Ha caratteristiche aritmetiche di base e una raccolta rudimentale di calcolo scientifico.

5. SQLAlchemy

Ha sofisticate funzionalità matematiche e SQLAlchemy è una libreria di programmazione matematica di base con tendenze ben note a livello aziendale. È stato creato per rendere la disponibilità del database efficiente e ad alte prestazioni.

6. Bella zuppa

Questo potrebbe essere un po' lento. BeautifulSoup sembra avere una superba libreria per l'analisi XML e HTML per principianti.

7. Piglet

Pyglet è una scelta eccezionale quando si progettano giochi con un'interfaccia del linguaggio di programmazione orientata agli oggetti. Vede anche l'uso nello sviluppo di altri programmi visivamente ricchi per Mac OS X, Windows e Linux in particolare. Negli anni '90, hanno iniziato a giocare a Minecraft sui loro PC ogni volta che le persone si annoiavano. Pyglet è il meccanismo che alimenta Minecraft.

8. SciPy

Il prossimo disponibile è SciPy, una delle librerie di cui abbiamo parlato così spesso. Ha una gamma di routine numeriche facili da usare ed efficaci. Questi forniscono routine di ottimizzazione e procedure di integrazione numerica.

9. Scrapy

Se il tuo obiettivo è rapido, raschiare il monitor di alto livello e scansionare la rete, scegli Scrapy. Può essere utilizzato per attività di raccolta dati per il monitoraggio e l'automazione dei test.

10. PyGame

PyGame offre un'interfaccia incredibilmente semplice per le librerie grafiche, audio e di input indipendenti dal sistema della Popular Direct Media Library (SDL).

11. Pitone contorto

Twisted è una libreria di rete basata su eventi utilizzata in Python e autorizzata con la licenza open source del MIT.

12. Cuscino

Pillow è un fork amichevole di PIL (Python Imaging Library) ma è più efficiente per l'utente. Il cuscino è il tuo migliore amico quando lavori con le immagini.

13. pywin32

Come suggerisce il nome, questo fornisce metodi e classi utili per interagire con Windows.

14. wxPython

Per Python, è un wrapper per wxWidgets.

15. iPython

iPython Python Library fornisce un'architettura di elaborazione distribuita parallela. Lo utilizzerai per creare, eseguire, testare e tenere traccia della programmazione parallela e distribuita.

16. Naso

Il naso fornisce l'esplorazione dei test alternativi e l'esecuzione dei processi di automazione dei test. Questo intende imitare il più possibile il comportamento del py.test.

17. Pallone

Flask è un framework web, con un piccolo core e diverse estensioni.

18. SymPy

È una libreria di matematica simbolica open-source. SymPy è un Computer Algebra System (CAS) completo con un codice molto semplice e di facile comprensione che è altamente espandibile. È implementato in Python e quindi non sono necessarie librerie esterne.

19. Tessuto

Oltre ad essere una libreria, Fabric è uno strumento da riga di comando per semplificare l'uso di SSH per i programmi di installazione o le attività di gestione della rete. È possibile eseguire la riga di comando locale o remota, caricare/scaricare file e persino richiedere l'ingresso dell'utente in corso o interrompere l'attività con esso.

20. PyGTK

PyGTK ti consente di creare programmi facilmente utilizzando una GUI (Graphical User Interface) Python.

Operatori ed espressioni

operatori

In Python, gli operatori sono simboli speciali che eseguono il calcolo di operazioni matematiche. Il valore su cui è in esecuzione l'operatore è chiamato operando.

Operatori aritmetici

Viene utilizzato dagli operatori aritmetici per eseguire operazioni matematiche come addizione, sottrazione, moltiplicazione, ecc.

Operatori di confronto

Gli operatori di confronto possono essere per confronti di valore A seconda delle condizioni, ritorna utilizzati True o False.

Operatori logici

Gli operatori logici sono and, or, not.

Operatore	Significato	Esempio
e	Vero se entrambi gli operandi sono veri	x e y
o	Vero se uno degli operandi è vero	x o y
Non	Vero se l'operando è falso (complementa l'operando)	non x

Operatori bit a bit

Gli operatori bit a bit funzionano come se diventassero stringhe di cifre binarie sugli operandi. A poco a poco funzionano,

e quindi il nome. Ad esempio, in binario due è 10 e in binario sette è 111.

Operatori di assegnazione

Gli operatori di assegnazione del linguaggio Python vengono utilizzati per assegnare valori alle variabili. `a = 5` è un semplice task operator che assegna il valore 'a' di 5 a destra della variabile 'a' a sinistra. In Python, ci sono vari operatori composti come `a += 5`, che aggiunge alla variabile e assegna la stessa in seguito. Questo è uguale a `a = a + 5`.

Operatori speciali

Il linguaggio Python fornisce altri diversi tipi di operatori, come l'operatore dell'identità o l'operatore dell'appartenenza. Esempi di questi sono menzionati di seguito.

Operatori di identità

'Is' e 'is not' sono operatori di identità Python. Sono usati per verificare se ci sono due valori o variabili nella stessa sezione di memoria. Due variabili uguali non significano che siano equivalenti.

Operatore di appartenenza

Gli operatori utilizzati per verificare se esiste o meno un valore/variabile nella sequenza come stringa, elenco, tuple, set e dizionario. Questi operatori restituiscono True o False se una variabile viene trovata nell'elenco, restituisce True, oppure restituisce False

espressioni

Un'espressione è un insieme di valori, variabili, operatori e chiamate di funzioni. Ci deve essere una valutazione delle espressioni. Quando chiedi a Python di stampare una frase, l'interprete valuterà l'espressione e mostrerà l'output.

Conversioni aritmetiche

Ogni volta che l'interpretazione di un operatore aritmetico di seguito utilizza la frase "gli argomenti numerici vengono convertiti in un tipo comune", ciò significa che l'esecuzione dell'operatore per le modalità integrate funziona come segue

Se un argomento è una quantità complessa, l'altro viene convertito in un numero complesso; Se un altro argomento è un numero a virgola mobile, l'altro argomento viene trasformato in un numero a virgola mobile; Oppure entrambi saranno numeri interi senza bisogno di conversione.

Atomi

Gli atomi sono i componenti espressivi più importanti. Gli atomi più piccoli sono letterali o identità astratte. Le forme contenute tra parentesi, parentesi quadre o parentesi graffe sono anche sintatticamente note come atomi. La sintassi degli atomi è:

atomo ::= identificatore | custodia | letterale

allegato ::= list_display | parenth_form | dict_display | set_display

Identificatori (nomi)

Un nome è un identificatore che si presenta come un atomo. Vedere la sezione Identificatori e parole chiave di descrizione lessicale e gruppo Denominazione e rilegatura per denominare e rilegare i documenti. Ogni volta che il nome è collegato a un'entità, produce l'entità valutando l'atomo. Quando un nome non è connesso, un tentativo di valutarlo eleva l'eccezione per NameError.

letterali

Python fornisce stringhe logiche e byte e letterali numerici di diversi tipi: literal::=

stringa literal | byte letterali

| intero | numero float | numero di immagine

Valutazione di una resa letterale con l'insieme previsto di un oggetto di quel tipo (byte, intero, numero a virgola mobile, stringa, numero complesso). Nello scenario di letterali a virgola mobile e immaginari (complessi), il valore può essere approssimato.

Forme tra parentesi

Un tipo tra parentesi è un insieme di parentesi disponibile per l'espressione:

```
parenth_form ::= "(" [starred_expression] ")"
```

Un elenco di espressioni tra parentesi restituisce tutto ciò che l'elenco di espressioni produce: se l'elenco include almeno una virgola, produce una tupla. In caso contrario, restituisce l'unica espressione che costituisce l'elenco di espressioni. Una coppia di parentesi nulla genera un oggetto incompleto di tuple. Poiché tutte le tuple sono immutabili, si applicano le stesse regole dei letterali (ovvero, due occorrenze di tuple vuote producono o meno la stessa entità).

Display per elenchi, set e dizionari

Per la costruzione di un elenco, Python utilizza una serie o un dizionario con una sintassi particolare chiamata "display", ciascuno in fili complementari:

I contenuti del contenitore sono elencati in modo esplicito, oppure sono calcolati utilizzando una serie di istruzioni per il ciclo e il filtraggio, denominate 'comprensione'. Le caratteristiche comuni della sintassi per la comprensione sono:

```
comprensione ::= assegnazione_espressionecompr_for compr_for ::=  
["async"] "for" target_list "in" or_test [comp_iter]  
comp_iter ::= compr_for | compr_if compr_if ::= "if"  
expression_nocond [comp_iter]
```

Una comprensione contiene una singola frase pronta per almeno un'espressione per clausola e zero o più clausole per o se. In tutta questa situazione, i componenti del contenitore sono quelli che verranno generati assumendo ciascuna delle clausole for o if come un blocco, nidificando da sinistra a destra e determinando la fase per la creazione di un'entità ogni volta che si avvicina il blocco del nucleo interno .

Visualizza l'elenco

Una vista elenco è una sequenza probabilmente vuota di parentesi quadre che include le espressioni: `list_display ::= "[" [starred_list | comprensione] "]"`

Una visualizzazione elenco genera un nuovo oggetto colonna, con un elenco di espressioni o una comprensione che specifica gli elementi. Quando viene fornito un database di espressioni separato da virgole, i suoi elementi vengono valutati da sinistra a destra e posizionati in quell'ordine nell'entità categoria. Quando viene fornita la comprensione, l'elenco deve essere costruito dai componenti della comprensione.

Imposta display

Le parentesi graffe denotano una visualizzazione impostata e possono essere distinte dalle visualizzazioni del dizionario per la mancanza di due punti che dividono i tipi di dati:

```
set_display ::= "{" (starred_list | comprensione) "}"
```

Uno spettacolo insieme offre una nuova entità insieme mutevole, con una serie di espressioni o una comprensione che definisce i contenuti. Quando viene fornito con un elenco di espressioni separato da virgole, i suoi elementi vengono valutati da sinistra a destra e assegnati all'entità impostata. Ogni volta che viene fornita una comprensione, l'insieme è formato dagli elementi derivati dalla comprensione. Impossibile creare un insieme vuoto con questo {}; letterale forma un dizionario vuoto.

Il dizionario visualizza

Una vista del dizionario è una sequenza potenzialmente vuota di coppie di chiavi limitate alle parentesi graffe:

```
dict_display ::= "{" [key_datum_list | dict_comprehension] "}"
key_datum_list ::= key_datum ("," key_datum)* [","]
key_datum ::= espressione ":" espressione | "***" o_espr
dict_comprehension ::= espressione ":" espressione comp_for
```

La vista del dizionario mostra un nuovo oggetto nel dizionario. Quando viene fornita una serie di coppie chiave/datum separate da virgole, queste vengono analizzate da sinistra a destra per identificare le voci del dizionario: ogni entità chiave viene spesso utilizzata come chiave per contenere il rispettivo dato nel dizionario. Ciò implica che è possibile indicare chiaramente la stessa chiave numerose volte nel catalogo chiavi/datum, ma l'ultima data diventerà il valore del dizionario finale per quella chiave.

Espressioni generatore

Un'espressione del generatore è la sintassi compressa di un generatore tra parentesi:

```
generator_expression ::= "(" expression comp_for ")"
```

Un generatore di espressioni produce un'entità che è un nuovo generatore. La sua sintassi sarà la stessa delle comprensioni, tranne per essere racchiusa tra parentesi o parentesi graffe anziché tra parentesi. Le variabili utilizzate dall'espressione del generatore vengono valutate in modo approssimativo quando l'oggetto generatore (nello stesso stile dei generatori standard) viene chiamato dal metodo `__next__()`. Viceversa, l'espressione iterabile nella parte più a sinistra della clausola viene valutata immediatamente, in modo tale che un errore che essa produce venga trasmesso al livello in cui è caratterizzata l'espressione del generatore, piuttosto che al livello in cui viene recuperato il primo valore .

Ad esempio: `(x*y per x in range(10) per y in range(x, x+10)).`

Espressioni di rendimento

```
rendimento_atomo ::= "(" espressione_rendimento ")"
yield_expression ::= "rendimento" [expression_list | "da" espressione]
```

L'espressione prodotta viene utilizzata per definire una funzione generatore o una funzione generatore asincrona e può quindi essere utilizzata solo nel corpo della definizione della funzione. L'uso di un'espressione di `yield` nel corpo di una funzione tende a far sì che tale funzione sia un generatore e l'utilizzo nel corpo di una funzione `def` asincrona induce tale funzione di co-routine a diventare un generatore asincrono. Per esempio:

```
def gen(): # definisce una funzione generatrice
    yield 123
asyncdef agen(): # definisce una funzione generatore asincrona
    yield 123
```

A causa dei loro effetti negativi sull'ambito di trasporto, le espressioni di resa non sono consentite come parte degli ambiti implicitamente definiti utilizzati per imporre comprensioni ed espressioni di generatori.

Input e output di dati in Python

Output Python utilizzando la funzione print()

Per visualizzare i dati nel sistema di visualizzazione standard (schermo), utilizziamo la funzione `print()`. Potremmo anche inviare dati a un server, ma questo verrà affrontato in seguito. Di seguito è riportato un esempio del suo utilizzo.

```
> > > print('Questa frase viene visualizzata sullo schermo')
```

Produzione:

Questa frase viene emessa sullo schermo

Viene fornito un altro esempio:

```
a = 5
```

```
print('Il valore di a è,' a)
```

Output:

Il valore di a è 5

All'interno della seconda dichiarazione di `print()`, noteremo che è stato inserito uno spazio tra la stringa e il valore della variabile `a`. Per impostazione predefinita, contiene questa sintassi, ma possiamo cambiarla.

La sintassi effettiva della funzione `print()` sarà:

```
print(*objects, sep=' ', end='\n', file=sys.stdout, flush=False)
```

Qui, l'oggetto è il valore o i valori che verranno stampati. Viene utilizzato il separatore `sep` tra i valori. Questo si trasforma in un carattere nello spazio. Dopo aver stampato tutti i valori, viene stampata la finitura. Si sposta in una nuova sezione in base alla progettazione. Il file è l'oggetto che stampa i valori e il suo valore predefinito è `sys.stdout` (schermo). Di seguito è riportato un esempio di ciò.

```
print(1, 2, 3, 4) print(1, 2, 3, 4,  
sep='*') print(1, 2, 3, 4, sep='#',  
end='&') Esegui codice
```

Produzione:

```
1 2 3 4  
1*2*3*4  
1#2#3#4&
```

Formattazione dell'output

Spesso vogliamo dare uno stile alla nostra produzione, quindi sembra attraente. Può essere fatto usando il metodo `str.format()`. Questa tecnica è visibile per qualsiasi oggetto con una stringa.

```
> > > x = 5; y = 10  
> > > print('Il valore di x è {} e y è {}'.format(x,y))
```

Qui il valore di `x` è cinque e `y` è 10

Qui, usano le parentesi graffe `{}` come sostituti. Usando i numeri (indice di tuple), possiamo specificare l'ordine in cui sono stati stampati.

```
print('Amo {0} e {1}'.format('bread','butter'))  
print('Amo {1} e {0}'.format('bread','butter'))
```

Esegui codice

Produzione:

```
Amo il pane e il burro  
Amo il burro e il pane
```

Le persone possono anche utilizzare argomenti con parole chiave per formattare la stringa.

```
> > > print('Ciao {nome}, {saluto}'.format(saluto = 'Buongiorno', nome = 'Giovanni'))  
Ciao Giovanni, buongiorno
```

A differenza del vecchio stile `sprint()` utilizzato nel linguaggio di programmazione C, possiamo anche formattare le stringhe. Per fare ciò, usiamo l'operatore `'%'`.

```
> > > x = 12.3456789
> > > print('Il valore di x è %3.2f' %x)
Il valore di x è 12,35
> > > print('Il valore di x è %3.4f' %x)
Il valore di x è 12,3457
```

Indentazione Python

Il rientro si applica agli spazi all'inizio di una riga del compilatore. Considerando che l'indentazione nel codice è per la leggibilità solo in altri linguaggi di programmazione, ma l'indentazione in Python è molto importante. Python supporta il rientro per denotare un blocco di codice.

Esempio

```
se 5 > 2:
    print("Cinque è maggiore di due!")
```

Python genererà un messaggio di errore se salti il rientro:

Esempio

Errore di sintassi:

```
se 5 > 2:
    print("Cinque è maggiore di due!")
```

Input Python

I nostri programmi sono stati statici. Le variabili sono state descritte o codificate nel codice sorgente. Vorremmo prendere il feedback dall'utente per consentire la flessibilità. Abbiamo la funzione `input()` in Python per abilitarlo. `input()` è sintassi come:

```
input([prompt])
```

Mentre `prompt` è la stringa che vogliamo mostrare sul computer, questo è facoltativo.

```
> > > num = input('Inserisci un numero: ')
```

Inserisci un numero: 10

```
> > > numero
```

```
'10'
```

Di seguito, possiamo vedere come il valore 10 inserito sia una stringa e non un numero. Per trasformarlo in un numero possiamo usare le funzioni `int()` o `float()`.

```
> > > int('10')
```

```
10
```

```
> > > float('10')
```

```
10.0
```

Lo stesso metodo può essere eseguito con la funzione `eval()`. Anche se `eval` richiede molto di più. Può anche quantificare le espressioni, a condizione che l'input sia una stringa

```
> > > int('2+3')
```

Traceback (ultima chiamata più recente):

File "<string>", riga 301, in runcode

File "<input interattivo>", riga 1, in <module>

ValueError: int() base 10 letterale non valido: '2+3'


```
> > > eval('2+3')
```

```
5
```

Importazione Python

Man mano che il nostro software diventa più grande, suddividerlo in moduli separati è un'idea intelligente. Un modulo è un file che contiene definizioni e istruzioni da Python. I pacchetti Python hanno un nome file e l'estensione .py inizia con esso. Le definizioni possono essere caricate in un altro modulo o nell'interprete Python integrato all'interno di un modulo. Per fare ciò, usiamo la parola chiave sull'importazione.

Ad esempio, scrivendo la riga sotto, possiamo importare il modulo math:

```
import math
```

Useremo il modulo come segue:

```
import math
```

```
print(math.pi)
```

Esegui codice

Produzione

```
3.141592653589933
```

Finora, tutti i concetti sono inclusi nel nostro framework all'interno del modulo di matematica. Gli sviluppatori possono anche importare solo determinati attributi e funzioni particolari, utilizzando la parola chiave.

Ad esempio:

```
> > > da importazione matematica pi
```

```
> > > pi
```

```
3.141592653589933
```

Python esamina più posizioni specificate in sys.path durante l'importazione di un modulo. È un elenco di posizioni in una directory.

```
> > > importa sistema
```

```
> > > sys.path
```

```
['',
```

```
'C:\\Python33\\Lib\\idlelib', 'C:\\
```

```
\\Windows\\system32\\python33.zip', 'C:\\
```

```
\\Python33\\DLL',
```

```
'C:\\Python33\\lib',
```

```
'C:\\Python33',
```

```
'C:\\Python33\\lib\\pacchetti-sito']
```

Possiamo inserire anche la nostra destinazione in quella lista.

FUNZIONI

SÌ Utilizzi le funzioni di programmazione per combinare un elenco di istruzioni che utilizzi costantemente o che sono meglio autocontenute nella complessità del sottoprogramma e vengono richiamate quando richiesto. Il che significa che una funzione è un tipo di codice scritto per raggiungere un determinato scopo. La funzione può richiedere o meno vari input per svolgere quel particolare compito. Ogni volta che l'attività viene eseguita, uno o più valori possono o non possono essere restituiti dalla funzione. Fondamentalmente esistono tre tipi di funzioni nel linguaggio Python:

1. Funzioni integrate, tra cui `help()` per chiedere aiuto, `min()` per ottenere solo la quantità minima, `print()` per stampare un attributo sul terminale. Altre di queste funzioni possono essere trovate qui.
2. Funzioni definite dall'utente (UDF) che sono funzioni create dagli utenti per assisterli e supportarli;
3. Funzioni anonime, denominate anche funzioni lambda poiché non sono definite con la parola chiave predefinita.

Definizione di una funzione: funzioni definite dall'utente (UDF)

I seguenti quattro passaggi servono per definire una funzione in Python:

1. La parola chiave `def` può essere utilizzata per dichiarare la funzione e quindi utilizzare il nome della funzione per tornare indietro.
2. Aggiungi parametri di funzione: devono essere tra parentesi di funzione. Termina la tua riga con i due punti.
3. Aggiungere istruzioni che dovrebbero essere implementate dalle funzioni.

Quando la funzione dovrebbe produrre qualcosa, termina la tua funzione con un'istruzione `return`. L'attività deve restituire un oggetto. Nessuno senza dichiarazione di restituzione. Esempio:

1. `def ciao():`
2. `print("Ciao mondo")`
3. ritorno

È ovvio che man mano che vai avanti, le funzioni diventeranno più complesse: puoi includere cicli `for`, controllo del flusso e altro per rendere le cose più dettagliate:

```
def ciao():  
    name = str(input("Inserisci il tuo nome: "))  
    if  
    name:  
    print ("Ciao " + str (nome))  
    altrimenti:  
    print("Ciao mondo")  
  
    Restituzione  
    Ciao()
```

Nella funzione sopra, stai chiedendo all'utente di dare un nome. Se non viene fornito alcun nome, verrà stampata la funzione "Hello World". In caso contrario, l'utente riceverà una frase "Hello" personalizzata. Inoltre, considera di poter specificare uno o più parametri per la funzione UDF. Quando parlerai del segmento Dichiarazioni sulle caratteristiche, ne ascolterai di più. Di conseguenza, come risultato della tua funzione, puoi o meno restituire uno o più valori.

La dichiarazione di ritorno

Nota che poiché stamperai qualcosa del genere nel tuo ciao) (UDF, non devi davvero restituirlo. Non ci sarà distinzione tra la funzione sopra e questa:

Esempio:

1. `defhello_noreturn():`
2. `print("Ciao mondo")`

Anche così, se desideri continuare a lavorare con il risultato della tua funzione e provare alcune altre funzioni su di esso, dovrai utilizzare l'istruzione `return` per restituire semplicemente un valore, come una stringa, un numero intero. Controlla il seguente scenario in cui `hello()` restituisce una stringa "hello" mentre la funzione `hello_noreturn()` restituisce `None`:

1. `def ciao():`
2. `print("Ciao mondo")`
3. `ritorno("ciao")`
4. `defhello_noreturn():`
5. `print("Ciao mondo")`
6. `# Moltiplica l'output di `hello()` con 2`
7. `ciao() * 2`
8. `# (Prova a) moltiplicare l'output di `hello_noreturn()` con 2`
9. `ciao_noreturn() * 2`

La parte secondaria ti dà un errore perché, con Nessuno, non puoi eseguire alcuna operazione. Otterrai un `TypeError` che sembra dire che `NoneType` (il `None`, che è il risultato di `hello_noreturn()`) e `int` (2) non possono eseguire l'operazione di moltiplicazione. Le funzioni Tip escono istantaneamente quando viene trovata un'istruzione `return`, anche se ciò significa che non restituiranno alcun risultato:

1. `def run():`
2. `per x nell'intervallo (10):`
3. `se x == 2:`
4. `ritorno`
5. `print("Esegui!")`
6. `eseguire()`

Un altro fattore degno di nota quando si ha a che fare con l'"espressione di ritorno" è che molti valori possono essere restituiti utilizzandola. Prendi in considerazione l'uso di tuple per questo. Ricordiamo che questa struttura dati è molto paragonabile a quella di una lista: può contenere valori diversi. Anche così, le tuple sono immutabili, il che significa che non puoi alterare alcuna quantità memorizzata in essa! Lo costruisci con l'aiuto delle doppie parentesi). Con l'aiuto della virgola e dell'operatore di assegnazione, puoi disassemblare le tuple in diverse variabili.

Leggi l'esempio seguente per capire come più valori possono essere restituiti dalla tua funzione:

1. `# Definisci `plus()`

2. `def più(a,b):`

3. `somma = a + b`

4. `return (somma, a)`

5. `# Chiama più() e scompatta le variabili`

6. `somma, a = più(3,4)`

7. `# Stampa sum()`

8. `print(somma)`

Si noti che la somma dell'istruzione `return, 'a'` risulterà esattamente come la restituzione (`sum, a`): la prima impacchetta semplicemente `total` e `a` in una tupla sotto il cofano!

Come chiamare una funzione

Hai già visto molti esempi nelle sezioni precedenti di come si può chiamare una funzione. Provare a chiamare una funzione significa eseguire la funzione che hai descritto, direttamente dal prompt di Python o tramite una funzione diversa (come hai visto nella parte "Funzioni nidificate"). Chiama la tua nuova funzione `hello()` aggiunta essenzialmente implementando `hello()` come nel blocco DataCamp Light come segue:

1. `ciao()`

Aggiunta di stringhe di documentazione alle funzioni Python

Ulteriori punti preziosi delle funzioni di scrittura di Python: docstrings. Le stringhe di documentazione definiscono cosa fa la tua funzione, come gli algoritmi che conduce o i valori che restituisce. Queste definizioni fungono da metadati per la tua funzione in modo che chiunque legga la docstring della tua funzione possa capire cosa sta facendo la tua funzione, senza dover seguire tutto il codice nella specifica della funzione. Le docstring delle attività sono posizionate subito dopo l'intestazione della funzione nella riga successiva e sono impostate tra virgolette triple. Per la tua funzione `hello()`, una docstring adatta è "Hello World print".

`def ciao():`

`"""Stampa "Hello World".`

`Restituisce:`

`Nessuno`

`"""`

`print("Ciao mondo")`

`Restituzione`

Nota che puoi estendere le docstring più di quella fornita qui come esempio. Se vuoi studiare le docstring in modo più approfondito, dovresti provare a dare un'occhiata ad alcuni repository Github della libreria Python come `scikit-learn` o `pandas`, in cui troverai molti buoni esempi!

Argomenti delle funzioni in Python

Probabilmente hai imparato prima la distinzione tra definizioni e affermazioni. In parole povere, gli argomenti sono gli aspetti che vengono dati a qualsiasi funzione o chiamata di metodo, mentre le loro identità di parametro rispondono agli argomenti nel codice della funzione o del metodo. Le UDF Python possono richiedere quattro tipi di argomenti:

1. Argomenti predefiniti

2. Argomenti richiesti

3. Argomenti delle parole chiave

4. Numero variabile di argomenti

Argomenti predefiniti

Gli argomenti predefiniti sarebbero quelli che accettano i dati predefiniti se non viene fornito alcun valore dell'argomento durante la funzione di chiamata. Con l'operatore di assegnazione =, come nel caso seguente, puoi assegnare questo valore predefinito:

1. #Definisci la funzione `plus()`
2. def più(a,b = 2):
- 3.return a + b
4. # Chiama `plus()` con solo `a` parametro
5. più(a=1)
6. # Chiama `plus()` con `a` e `b` parametri
7. più(a=1, b=3)

Argomenti richiesti

Poiché il nome fa emergere, le affermazioni di cui ha bisogno un UDF sono quelle che ci saranno. Tali istruzioni devono essere trasferite durante la chiamata di funzione e sono assolutamente nell'ordine corretto, come nell'esempio seguente:

1. # Definisci `plus()` con gli argomenti richiesti
2. def più(a,b):
3. restituire a + b

Chiamando le funzioni senza ricevere errori aggiuntivi, sono necessari argomenti che mappano su 'a' così come i parametri 'b'. Il risultato non sarà univoco se si scambiano "a" e "b", ma potrebbe esserlo se si modifica plus() come segue:

1. # Definisci `plus()` con gli argomenti richiesti
2. def più(a,b):
- 3.return a/b

Argomenti delle parole chiave

Utilizzerai gli argomenti delle parole chiave nella chiamata di funzione se desideri assicurarti di elencare tutti i parametri nell'ordine corretto. Si usa per definire le istruzioni in base al nome della funzione. Facciamo un esempio sopra per renderlo un po' più semplice:

1. # Definisci la funzione `plus()`
2. def più(a,b):
- 3.return a + b
4. # Chiama la funzione `plus()` con parametri
5. più (2,3)
6. # Chiama la funzione `plus()` con argomenti chiave
7. più(a=1, b=2)

Nota che puoi anche modificare la sequenza dei parametri utilizzando gli argomenti delle parole chiave e ottenere comunque lo stesso risultato durante l'esecuzione della funzione:

1. # Definisci la funzione `plus()`
2. def più(a,b):
- 3.return a + b

4. # Chiama la funzione `plus()` con argomenti chiave

5. più(b=2, a=1)

Variabili globali e variabili locali

Le variabili identificate all'interno di una struttura di funzione di solito hanno un ambito locale e quelle specificate all'esterno hanno un ambito globale. Ciò mostra che le variabili locali sono specificate all'interno di un blocco funzione e possono essere recuperate solo tramite tale funzione, mentre le variabili globali possono essere recuperate da tutte le funzioni nella codifica:

1. # Variabile globale `init`

2. init = 1

3. # Definisci la funzione `plus()` per accettare un numero variabile di argomenti

4. def plus(*args):

5. # Variabile locale `sum`()

6. totale = 0

7. for i in argomenti:

8. totale += i

9. ritorno totale

10. # Accedi alla variabile globale

11. print("questo è il valore inizializzato " + str(init))

12. # (Prova a) accedere alla variabile locale

13. print("questa è la somma " + str(totale))

Scoprirai che puoi ottenere un `NameError` che significa che il nome "total" non è specificato mentre tenti di stampare la variabile locale totale che è stata specificata nel corpo della funzione. In confronto, l'attributo `init` può essere scritto senza complicazioni.

Funzioni anonime in Python

Le funzioni anonime sono spesso chiamate funzioni lambda in Python poiché si utilizza la parola chiave `lambda` invece di nominarla con la parola chiave standard-`def`.

1. doppio = lambda x: x*2

2. doppio(5)

La funzione anonima o lambda nel blocco DataCamp Light sopra è `lambda x: x*2`. `x` è l'argomento e `x*2` è l'interpretazione o l'istruzione che viene analizzata e restituita. Ciò che è unico in questa funzione, e non ha tag, come gli esempi che hai visto nella prima sezione della lezione per questa funzione. Quando dovevi scrivere la funzione sopra in una UDF, otterresti il seguente risultato:

```
def doppia(x):
```

```
    ritorna x*2
```

Vediamo un altro esempio di funzione lambda in cui vengono utilizzati due argomenti:

1. # `sum()` funzione lambda

2. somma = lambda x, y: x + y;

3. # Chiama la funzione anonima `sum()`

4. somma(4,5)

5. # "Traduci" in una UDF

6. def somma(x, y):

7. ritornox+y

Quando hai bisogno di una funzione senza nome per un breve intervallo di tempo, utilizzi funzioni anonime e questa viene generata in fase di esecuzione. Contesti speciali in cui questo è importante sono quando si opera con filter(), map() e redu():

1. da functools import riduci

2. mia_lista = [1,2,3,4,5,6,7,8,9,10]

3. # Usa la funzione lambda con `filter()

4. filtered_list = list(filter(lambda x: (x*2 > 10), my_list))

5. # Usa la funzione lambda con `map()

6. mapped_list = list(map(lambda x: x*2, my_list))

7. # Usa la funzione lambda con `reduce()

8. lista_ridotta = riduci(lambda x, y: x+y, mia_lista)

9. print(lista_filtrata)

10. print(lista_mappata)

11. print(lista_ridotta)

Poiché il nome indica la funzione filter(), aiuta a filtrare l'elenco originale di input my_list in base a un criterio > 10. Al contrario, con map(), implementi una funzione per ogni componente in my_listlist. In questo scenario moltiplichi tutti i componenti per due. Ricorda che la funzione reduce() è una parte della libreria functools. Si utilizza cumulativamente questa funzione per i componenti nell'elenco my_list(), da sinistra a destra, e in questa situazione si riduce la sequenza a un singolo valore 55.

Utilizzo di main() come funzione

Se hai qualche conoscenza con altri linguaggi di programmazione come java, noterai che l'esecuzione delle funzioni richiede la funzionalità principale. Come hai saputo negli esempi precedenti, Python non lo richiede davvero. Tuttavia, può essere utile organizzare logicamente il tuo codice insieme a una funzione main() nel tuo codice python: tutti i componenti più importanti sono contenuti all'interno di questa funzione main().

Potresti anche semplicemente ottenere e chiamare una funzione main() come hai fatto con tutte quelle funzioni sopra:

1. # Definisci la funzione `main()`

2. def principale():

3. ciao()

4. print("Questa è la funzione principale")

5. principale()

Dopotutto, come appare ora, quando lo carichi come modulo, verrà effettivamente chiamato lo script della tua funzione principale (). Invoca la funzione main() ogni volta che name == 'main' per assicurarvi che ciò non accada.

Ciò implica che lo script del codice sorgente sopra diventa:

1.# Definisci la funzione `main()`

2.def principale():

3.ciao()

4.print("Questa è una funzione principale")

5.# Esegui la funzione `main()`

6. if __name__ == '__main__':

7. principale()

Ricorda che oltre alla funzione main, hai anche una funzione init, che convalida una classe o un'istanza di oggetto. Definito in modo chiaro, funziona come un costruttore o un inizializzatore e viene definito automaticamente quando si avvia una nuova istanza di classe. Con una tale funzione, l'oggetto appena formato viene assegnato al parametro self che hai visto in precedenza in questa guida.

Considera il seguente esempio:

class Dog:

"""

Richiede:

gambe – gambe per far camminare un cane.

colore – Colore della pelliccia.

"""

def __init__(self, gambe, colore):

self.legs = gambe

self.color = colore

def corteccia (te stesso):

abbaia = "abbaia" * 2

restituisce abbaia

if __name__ == "__main__":

cane = Cane(4, "marrone")

abbaia = cane.abbaia()

stampa (corteccia)

LISTE E LOOP

Elenchi

UN list è spesso una struttura dati in Python, che è un elenco ordinato di elementi che è mutabile o modificabile. Un elemento è denominato per ogni elemento o valore all'interno di un elenco. Proprio come le stringhe sono definite come caratteri tra virgolette, gli elenchi sono specificati da parentesi quadre '[']' con valori.

Le liste sono belle da avere perché hai altri principi simili da affrontare. Ti aiutano a mantenere intatti i dati rilevanti, a comprimere il codice ed eseguire gli stessi metodi e processi a più valori contemporaneamente.

Potrebbe essere utile ottenere tutti i vari elenchi che hai sul tuo computer quando inizi a pensare agli elenchi Python e ad altre strutture di dati che sono tipi di raccolte: il tuo assemblaggio di file, playlist di brani, segnalibri del browser, e-mail, raccolte di video che puoi accedere tramite una piattaforma di streaming e molto altro.

Dobbiamo funzionare con questa tabella di dati, presa dalla raccolta dati dell'App Store mobile (Ramanathan Perumal):

Nome	prezzo	valuta	rating_count	valutazione
Instagram	0.0	Dollaro statunitense	2161558	4.5
Scontro tra clan	0.0	Dollaro statunitense	2130805	4.5
Corsa al tempio	0.0	Dollaro statunitense	1724546	4.5
Pandora – Musica e radio	0.0	Dollaro statunitense	1126879	4.0
Facebook	0.0	Dollaro statunitense	2974676	3.5

Ogni valore è un punto dati nella tabella. La prima riga (subito dopo i titoli delle colonne) ad esempio ha 5 punti dati:

- Facebook
- 0.0
- Dollaro statunitense
- 2974676
- 3.5

Il set di dati è costituito da una raccolta di punti dati. Possiamo considerare la tabella sopra come un elenco di punti dati. Pertanto consideriamo l'intero elenco un set di dati. Possiamo vedere che ci sono cinque righe e cinque colonne nel nostro set di dati.

Utilizzando la nostra conoscenza dei tipi Python, potremmo forse considerare di poter memorizzare ogni punto dati nella propria variabile - ad esempio, ecco come possiamo memorizzare i punti dati della prima riga:

script.py

```
track_name_row1 = 'Facebook'  
price_row1 = 0.0  
currency_row1 = 'USD'  
rating_count_tot_row1 = 2974676  
user_rating_row1 = 3.5
```

Sopra, abbiamo memorizzato:

- Testo per la stringa come "Facebook".
- Float 0.0 come prezzo
- Testo per la stringa come "USD". Intero 2.974.676
- come conteggio della valutazione Float 3.5 per la
- valutazione dell'utente

Un processo complicato sarebbe creare una variabile per ogni punto dati nel nostro set di dati. Fortunatamente possiamo utilizzare gli elenchi per archiviare i dati in modo più efficace. Quindi, nella prima riga, possiamo stilare un elenco di punti dati:

script.py

```
row_1 = ['Facebook', 0.0, 'USD', 2974676, 3.5]  
print(row_1)  
type(row_1)
```

Output

```
['Facebook', 0.0, 'USD', 2974676, 3.5]  
list
```

Per la creazione della lista, noi:

- Separare ciascuno con una virgola durante la digitazione di una sequenza di punti dati: "Facebook", 0.0, 'USD', 2974676, 3.5
- Chiusura dell'elenco con parentesi quadre: ['Facebook', 0.0, 'USD', 2974676, 3.5]
- Dopo che l'elenco è stato creato, lo assegniamo a una variabile denominata riga_1 e l'elenco viene archiviato nella memoria del computer.

Per creare un elenco di punti dati, abbiamo solo bisogno di:

- Aggiungi una virgola ai punti dati per la separazione.
- Chiudo l'elenco tra parentesi.

Vedi sotto mentre creiamo cinque elenchi, nel set di dati ogni riga con un elenco:

```
row_1 = ['FACEBOOK', 0.0, 'usd', 2974676, 3.5]

row_2 = ['INSTAGRAM', 0.0, 'usd', 2161558, 4.5]

row_3 = ['CLASH OF CLANS', 0.0, 'usd', 2130805, 4.5]

row_4 = ['TEMPLE RUN', 0.0, 'usd', 1724546, 4.5]

row_5 = ['PANDORA', 0.0, 'usd', 1126879, 4.0]
```

Indice delle liste Python

Un elenco potrebbe includere una gamma più ampia di tipi di dati. Un elenco contenente [4, 5, 6] include gli stessi tipi di dati (solo numeri interi), mentre l'elenco ['Facebook', 0.0, 'USD', 2974676, 3.5] contiene molti tipi di dati:

- Composto da due tipi di float (0.0, 3.5) Consistente da
- un tipo di intero (2974676) Composto da due tipi di
- stringhe ("Facebook", "USD")

```
['FACEBOOK', 0.0, 'usd', 2974676, 3.5]
```

l'elenco ha ottenuto 5 punti dati. Per la lunghezza di un elenco,

il comando len() può essere usato:

script.py

```
row_1 = ['Facebook', 0.0, 'USD', 2974676, 3.5]
print(len(row_1))

list_1 = [1, 6, 0]
print(len(list_1))

list_2 = []
print(len(list_2))
```

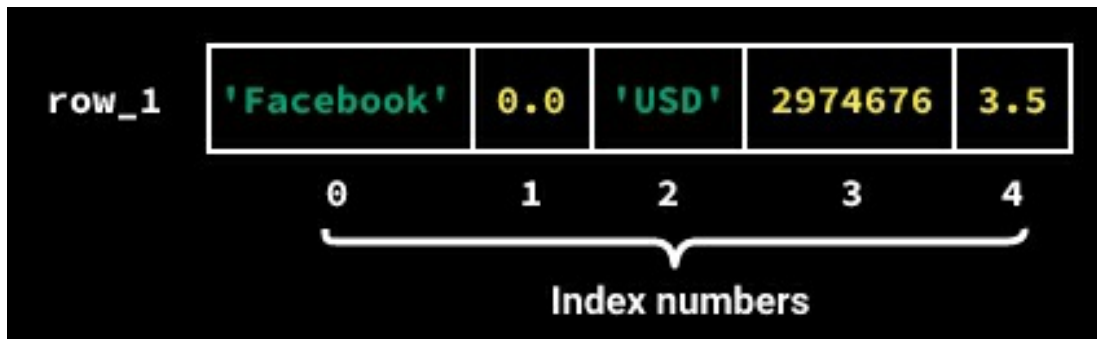
Output

```
5
3
0
```

Per elenchi più piccoli, possiamo semplicemente contare i punti dati sui nostri display per calcolare la lunghezza, ma

forse il comando `len()` dichiarerà di essere molto utile ogni volta che lavori con elenchi contenenti molti componenti, o semplicemente hai bisogno di comporre codice dati di cui non conosci davvero la lunghezza in anticipo.

Ogni altro componente (punto dati) in un elenco è collegato a un numero particolare, chiamato numero indice. Anche l'indicizzazione inizia da 0, il che significa che il primo elemento dovrebbe avere il numero di indice 0, il secondo elemento il numero di indice 1, ecc.



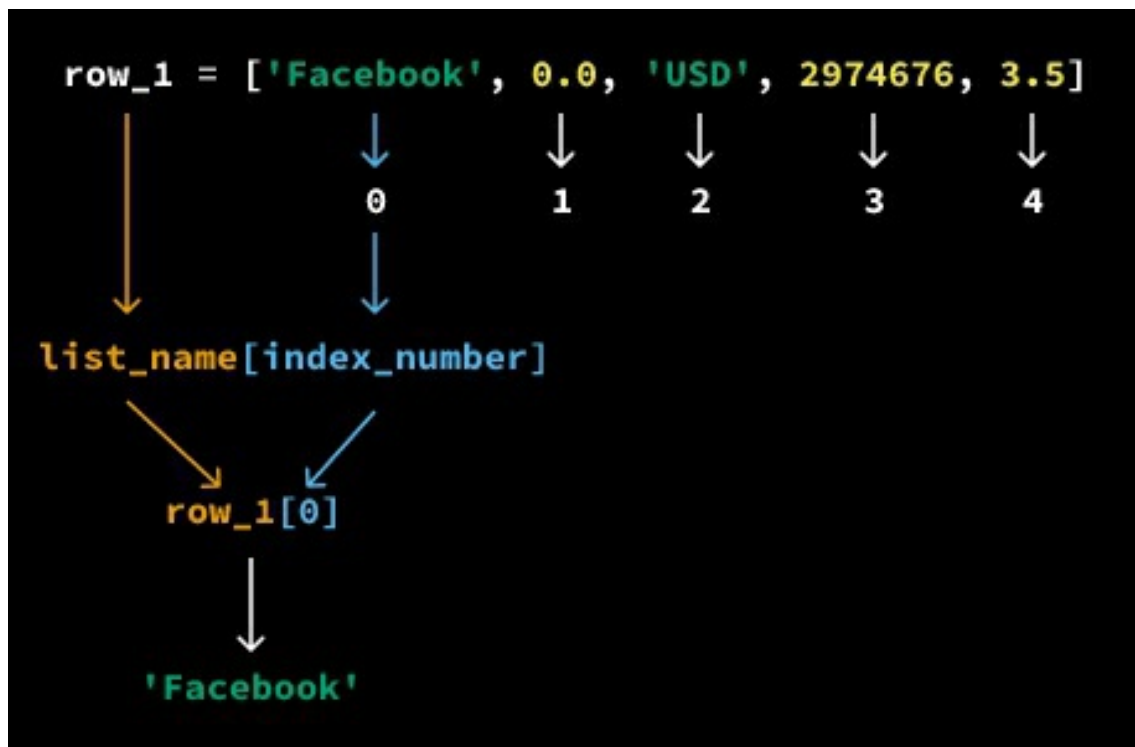
Per individuare rapidamente un indice di un elemento della lista, determina il suo numero di posizione nella lista e poi sottrailo per 1. La stringa "USD", ad esempio, è il terzo elemento nell'elenco (posizione numero 3), il suo numero di indice deve essere due perché $3 - 1 = 2$.

I numeri di indice ci consentono di individuare un singolo elemento da un elenco. Ripercorrendo a ritroso la riga 1 dell'elenco dell'esempio sopra, eseguendo la riga di codice `row_1[0]`, possiamo ottenere il primo nodo (la stringa 'Facebook') del numero di indice 0.

```
script.py
row_1 = ['Facebook', 0.0, 'USD', 2974676, 3.5]
row_1[0]

Output
'Facebook'
```

Il modello `list_name[numero indice]` segue la sintassi per individuare componenti di elenco specifici. Ad esempio, il titolo della nostra lista sopra è `row_1` e il numero indice di un primo elemento è 0, otteniamo `row_1[0]` continuando a seguire il modello `list_name[numero indice]`, in cui il numero indice 0 è solo tra parentesi quadre dopo il nome della variabile `row_1`.



Il metodo per recuperare ogni elemento nella riga_1:

```
script.py
row_1 = ['Facebook', 0.0, 'USD', 2974676, 3.5]
        0         1         2         3         4
        └──────────────────────────┘
                Index numbers

print(row_1[0])
print(row_1[1])
print(row_1[2])
print(row_1[3])
print(row_1[4])

Output
Facebook
0.0
USD
2974676
3.5
```

Il recupero degli elementi dell'elenco semplifica l'esecuzione dei processi. Ad esempio, è possibile selezionare le valutazioni di Facebook e Instagram e trovare l'aggregato o la distinzione tra i due:

```
script.py  
  
row_1 = ['Facebook', 0.0, 'USD', 2974676, 3.5]  
row_2 = ['Instagram', 0.0, 'USD', 2161558, 4.5]  
  
difference = row_2[4] - row_1[4]  
average_rating = (row_1[4] + row_2[4]) / 2  
  
print(difference)  
print(average_rating)
```

Output

```
1.0  
4.0
```

Prova a utilizzare l'indicizzazione dell'elenco per recuperare e quindi fare la media del numero di valutazioni con le prime 3 righe:

```
valutazioni_1 = riga_1[3]
```

```
valutazioni_2 = riga_2[3]
```

```
rating_3 = riga_3[3]
```

```
totale = voti_1 + voti_2 + voti_3
```

```
media = totale / 3
```

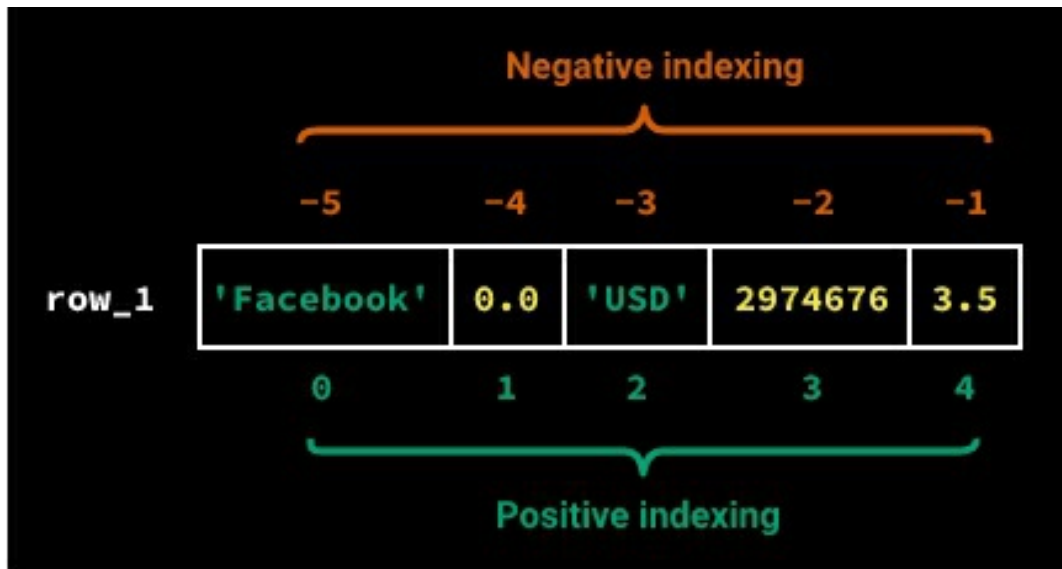
```
stampa (media)
```

```
2422346.3333333335
```

Utilizzo dell'indicizzazione negativa con gli elenchi

Esistono due sistemi di indicizzazione per gli elenchi in Python:

1. Indicizzazione positiva: il numero di indice del primo elemento è 0; il numero indice del secondo elemento è 1 e inoltre.
2. Indicizzazione negativa: il numero di indice dell'ultimo elemento è -1; il numero indice del secondo elemento è -2 e inoltre.



In esercizio, utilizziamo principalmente l'indicizzazione positiva per ottenere elementi della lista. L'indicizzazione negativa è utile ogni volta che vogliamo scegliere l'ultimo elemento in un elenco di questo tipo, soprattutto se l'elenco è lungo e, calcolando, non possiamo calcolare la lunghezza.

```
script.py  
  
row_1 = ['Facebook', 0.0, 'USD', 2974676, 3.5]  
  
print(row_1[-1])  
print(row_1[4])  
  
Output  
3.5  
3.5
```

Nota che quando usiamo un numero di indice appena al di fuori dell'ambito dei due schemi di indicizzazione, avremo un `IndexError`.

```
script.py
row_1 = ['Facebook', 0.0, 'USD', 2974676, 3.5]
row_1[6]

Output
IndexError: list index out of range
```

```
script.py
row_1 = ['Facebook', 0.0, 'USD', 2974676, 3.5]
row_1[-7]

Output
IndexError: list index out of range
```

Che ne dici di utilizzare l'indicizzazione negativa per rimuovere da ciascuna delle prime 3 righe la valutazione dell'utente (l'ultimo valore) e successivamente la media.

```
riga_1 [-1]=voto_1
riga_2[-1]=voto_2
riga_3[-1]=voto_3
voto_1 + voto_2 + voto_3= voto_totale
voto_totale / 3= voto_medio
stampa (media)
2422346.33333
```

Slice Python Liste

Piuttosto che selezionare gli elementi della lista separatamente, possiamo scegliere due elementi consecutivi usando una scorciatoia di sintassi:

script.py

```
row_3 = ['Clash of Clans', 0.0, 'USD', 2130805, 4.5]

cc_pricing_data = row_3[0:3] ← syntax shortcut
print(cc_pricing_data)
```

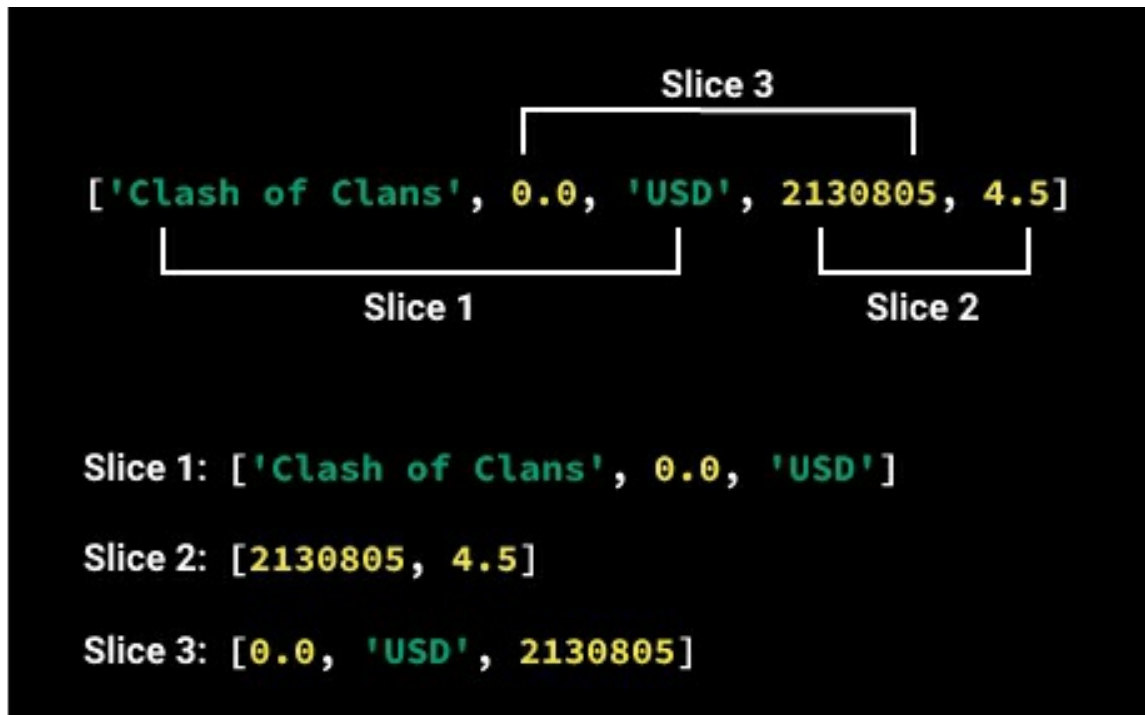
Output

```
['Clash of Clans', 0.0, 'USD']
```

Mentre selezioniamo i primi n elementi da una lista chiamata lista (n sta per un numero), possiamo usare la scorciatoia per la sintassi della lista $[0:n]$. Nell'esempio sopra, abbiamo dovuto scegliere dalla riga 3 dell'elenco i primi tre elementi, quindi useremo la riga $3[0:3]$.

Quando sono stati scelti i primi tre elementi, abbiamo affettato una parte del set. Per questa funzione, il metodo di raccolta per una sezione di un elenco è noto come affettare l'elenco.

La sezione dell'elenco può essere eseguita in molti modi:

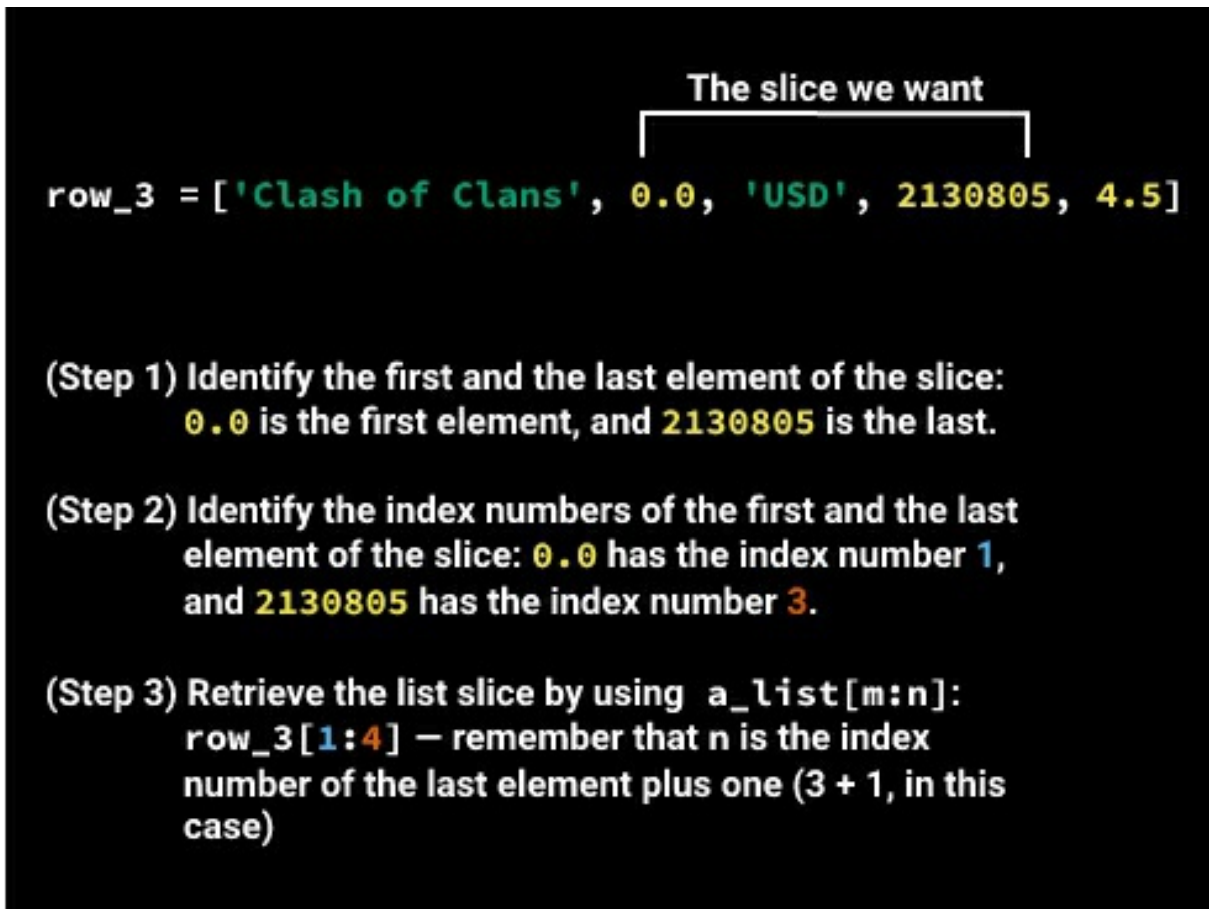


Recuperando qualsiasi fetta di elenco di cui abbiamo bisogno:

- Innanzitutto identificare il primo e l'ultimo elemento della fetta.
- Devono quindi essere definiti i numeri indice del primo e dell'ultimo elemento della fetta. Infine,
- possiamo usare la sintassi a list[m: n] per estrarre la slice di lista che ci occorre, mentre:

'm' indica il numero di indice di entrambi il primo elemento della fetta; e 'n' simboleggia il numero di indice dell'ultimo elemento della fetta oltre a uno (se l'ultimo elemento sembra avere il numero di indice 2, dopo di che

n è 3, se l'ultimo elemento sembra avere il numero di indice 4, dopo di che n è 5, e così via).



The slice we want

```
row_3 = ['Clash of Clans', 0.0, 'USD', 2130805, 4.5]
```

(Step 1) Identify the first and the last element of the slice:
0.0 is the first element, and 2130805 is the last.

(Step 2) Identify the index numbers of the first and the last element of the slice: 0.0 has the index number 1, and 2130805 has the index number 3.

(Step 3) Retrieve the list slice by using `a_list[m:n]`:
`row_3[1:4]` — remember that n is the index number of the last element plus one (3 + 1, in this case)

Quando vogliamo scegliere il primo o l'ultimo elemento 'x' (x rappresenta un numero), possiamo usare scorciatoie anche meno complesse per la sintassi:

`a_list[:x]` quando dobbiamo scegliere i primi x elementi. `a_list[-`

`x:]` quando dobbiamo scegliere gli ultimi x elementi.

script.py

```
row_3 = ['Clash of Clans', 0.0, 'USD', 2130805, 4.5]

first_3 = row_3[:3]
last_3 = row_3[-3:]

print(first_3)
print(last_3)
```

Output

```
['Clash of Clans', 0.0, 'USD']
['USD', 2130805, 4.5]
```

Guarda come recuperiamo dalla prima riga i primi quattro elementi (con i dati di Facebook):

```
first_4_fb = row_1[:4]
```

```
stampa(first_4_fb)
```

```
['Facebook', 0,0, 'USD', 2974676]
```

Dalla stessa riga, gli ultimi tre elementi sono:

```
last_3_fb = riga_1[-3:]
```

```
stampa(last_3_fb)
```

```
['USD', 2974676, 3.5]
```

Nella quinta riga (dati nella riga per Pandora) con gli elementi terzo e quarto sono:

```
pandora_3_4 = riga_5[2:4]
```

```
stampa(pandora_3_4)
```

```
['USD', 1126879]
```

Elenco di elenchi Python

Gli elenchi sono stati precedentemente introdotti come approccio praticabile all'utilizzo di una variabile per punto dati. Invece di avere una variabile diversa per uno qualsiasi dei cinque punti dati "Facebook", 0.0, "USD," 2974676, 3.5, possiamo collegare i punti dati in un elenco insieme e quindi salvare l'elenco in una variabile.

Da allora abbiamo lavorato con un set di dati di cinque righe e abbiamo archiviato ogni riga come raccolta in ciascuna variabile diversa (riga 1, riga 2, riga 3, riga 4 e riga 5 variabili). Anche così, se avessimo un set di dati di 5.000 righe, probabilmente avremmo finito con 5.000 variabili che creeranno il nostro codice disordinato e quasi difficile da lavorare.

Per risolvere questo problema, possiamo memorizzare le nostre cinque variabili in un elenco unificato:

script.py

```

row_1 = ['Facebook', 0.0, 'USD', 2974676, 3.5]
row_2 = ['Instagram', 0.0, 'USD', 2161558, 4.5]
row_3 = ['Clash of Clans', 0.0, 'USD', 2130805, 4.5]
row_4 = ['Temple Run', 0.0, 'USD', 1724546, 4.5]
row_5 = ['Pandora - Music & Radio', 0.0, 'USD', 1126879, 4.0]

data_set = [row_1, row_2, row_3, row_4, row_5]
data_set

```

Output**Notice the double brackets**

```

[['Facebook', 0.0, 'USD', 2974676, 3.5],
 ['Instagram', 0.0, 'USD', 2161558, 4.5],
 ['Clash of Clans', 0.0, 'USD', 2130805, 4.5],
 ['Temple Run', 0.0, 'USD', 1724546, 4.5],
 ['Pandora - Music & Radio', 0.0, 'USD', 1126879, 4.0]]

```

Notice the commas

Come vediamo, il set di dati è un elenco di cinque colonne aggiuntive (riga 1, riga 2, riga 3, riga 4 e riga 5). Un elenco contenente altri elenchi è definito un insieme di elenchi.

La variabile del set di dati è già una lista, il che indica che possiamo usare la sintassi che abbiamo imparato per recuperare i singoli elementi della lista ed eseguire lo slicing della lista. Sotto, abbiamo:

- Utilizzare `dataset[0]` per individuare il primo elemento dell'elenco (riga 1).
- Utilizzare `dataset[-1]` per individuare l'ultimo elemento dell'elenco (riga 5).
- Ottieni i primi due elementi dell'elenco (riga 1 e riga 2) utilizzando il set di dati `dataset[:2]` per eseguire una suddivisione dell'elenco.

script.py

```
data_set = [row_1, row_2, row_3, row_4, row_5]
print(data_set[0])
print(data_set[-1])
print(data_set[:2])
```

Output

```
['Facebook', 0.0, 'USD', 2974676, 3.5]
['Pandora - Music & Radio', 0.0, 'USD', 1126879, 4.0]
[['Facebook', 0.0, 'USD', 2974676, 3.5],
 ['Instagram', 0.0, 'USD', 2161558, 4.5]]
```

Spesso avremo bisogno di ottenere singoli elementi da un elenco che è una parte di un elenco di elenchi, ad esempio; potremmo aver bisogno di ottenere il rating di 3.5 dalla riga di dati ['FACEBOOK', 0.0, 'USD', 2974676, 3.5], che è una parte dell'elenco dei set di dati. Recuperiamo 3.5 dal set di dati di seguito utilizzando ciò che abbiamo appreso:

- Utilizzando data_set[0], individuiamo row_1 e allochiamo l'output a una variabile denominata fb_row.
- fb_row ['Facebook', 0.0, 'USD', 2974676, 3.5], che abbiamo stampato.
- Usando fb_row[-1], individuiamo l'elemento finale da fb_row (perché fb_row è una lista) e assegniamo l'output a una variabile chiamata fb_rating.
- Stampa fb_rating, output 3.5

script.py

```
data_set = [row_1, row_2, row_3, row_4, row_5]
fb_row = data_set[0]
print(fb_row)

fb_rating = fb_row[-1]
print(fb_rating)
```


Output

```
['Facebook', 0.0, 'USD', 2974676, 3.5]
3.5
```

In precedenza in questo esempio, abbiamo ottenuto 3.5 in due passaggi: prima è stato recuperato data_set[0], quindi è stato recuperato fb_row[-1]. C'è anche un modo semplice per ottenere lo stesso output 3.5 collegando i due indici ([0] e [-1]); il codice data_set[0][-1] ottiene 3.5.:

script.py

```
data_set = [row_1, row_2, row_3, row_4, row_5]
print(data_set[0][-1])
```



```
['Facebook', 0.0, 'USD', 2974676, 3.5]
```

Output

3.5

In precedenza in questo esempio, abbiamo visto due modi per recuperare il valore 3.5. Entrambi i metodi portano alle stesse prestazioni (3.5), ma il secondo approccio richiede meno codifica, poiché i passaggi che vediamo dall'esempio sono elegantemente integrati. Poiché è possibile selezionare un'alternativa, le persone generalmente preferiscono quest'ultima.

Trasformiamo i nostri cinque elenchi indipendenti nell'elenco di

elenchi: app_data_set = [row_1, row_2, row_3, row_4, row_5] quindi

utilizzare:

```
print(app_data_set)
```

```
[
    ['FACEBOOK', 0.0, 'usd', 2974676, 3.5]
    ['INSTAGRAM', 0.0, 'usd', 2161558, 4.5]
    ['CLASH OF CLANS', 0.0, 'usd', 2130805, 4.5]
    ['TEMPLE RUN', 0.0, 'usd', 1724546, 4.5]
    ['PANDORA', 0.0, 'usd', 1126879, 4.0]
]
```

Elenca i processi per metodo ripetitivo

In precedenza, eravamo interessati a misurare il posizionamento medio di un'app in questo progetto. Era un compito fattibile mentre stavamo cercando di lavorare solo per tre file, ma più diventa difficile, più ulteriori file aggiungiamo. Utilizzando la nostra tattica fin dall'inizio, noi:

1. Ottieni ogni valutazione individuale.
2. Prendi la somma delle valutazioni.
3. Dividendo per il numero totale di voti.

```
script.py
row_1 = ['Facebook', 0.0, 'USD', 2974676, 3.5]
row_2 = ['Instagram', 0.0, 'USD', 2161558, 4.5]
row_3 = ['Clash of Clans', 0.0, 'USD', 2130805, 4.5]
row_4 = ['Temple Run', 0.0, 'USD', 1724546, 4.5]
row_5 = ['Pandora - Music & Radio', 0.0, 'USD', 1126879, 4.0]

app_data_set = [row_1, row_2, row_3, row_4, row_5]
avg_rating = (app_data_set[0][-1] + app_data_set[1][-1] +
              app_data_set[2][-1] + app_data_set[3][-1] +
              app_data_set[4][-1]) / 5

avg_rating

Output
4.2
```

Come hai visto diventa complicato con cinque valutazioni. A meno che non avessimo a che fare con dati che includono migliaia di righe, sarebbe necessaria una quantità di codice inimmaginabile! Dovremmo trovare un modo rapido per recuperare un sacco di voti.

Dando un'occhiata all'esempio di codice in precedenza in questo thread, vediamo che una procedura continua a ripetere: all'interno di `app_data_set`, selezioniamo l'ultimo elemento della lista per ogni lista. E se potessimo semplicemente chiedere direttamente a Python che vorremmo ripetere questo processo in `app_data_set` per ogni elenco?

Fortunatamente possiamo usarlo: Python ci offre un percorso semplice per ripetere un piano che ci aiuta moltissimo quando dobbiamo ripetere un processo decine di migliaia o addirittura milioni di volte.

Supponiamo di avere una lista `[3, 5, 1, 2]` assegnata a una variabile `ratings`, e di dover replicare la seguente procedura: visualizzare l'elemento per ogni elemento nei `ratings`. Ed è così che possiamo trasformarlo in sintassi con Python:

```
script.py
ratings = [3, 5, 1, 2]

for element in ratings:
    print(element)

Output
3
5
1
2
```


La procedura che abbiamo deciso di replicare nel nostro primo esempio sopra è stata "generare l'ultimo elemento per ogni elenco nell'app_data_set". Ecco come possiamo trasformare quell'operazione in sintassi con Python:

script.py

```
app_data_set = [row_1, row_2, row_3, row_4, row_5]

for each_list in app_data_set:
    rating = each_list[-1]
    print(rating)
```

Output

```
3.5
4.5
4.5
4.5
4.0
```

Proviamo e poi ci facciamo un'idea di cosa sta succedendo sopra. Python differenzia ogni elemento dell'elenco da app_data_set, ciascuno alla volta, e lo assegna a each_list (che essenzialmente diventa un vettore che contiene un elenco - lo affronteremo ulteriormente):

script.py

```
app_data_set = [row_1, row_2, row_3, row_4, row_5]

for each_list in app_data_set:
    print(each_list)
```

Output

```
['Facebook', 0.0, 'USD', 2974676, 3.5]
['Instagram', 0.0, 'USD', 2161558, 4.5]
['Clash of Clans', 0.0, 'USD', 2130805, 4.5]
['Temple Run', 0.0, 'USD', 1724546, 4.5]
['Pandora - Music & Radio', 0.0, 'USD', 1126879, 4.0]
```

Nell'ultima figura in precedenza in questo thread, il codice è un'edizione molto più semplice e molto più concettuale del codice seguente:

script.py

```
app_data_set = [row_1, row_2, row_3, row_4, row_5]

print(app_data_set[0])
print(app_data_set[1])
print(app_data_set[2])
print(app_data_set[3])
print(app_data_set[4])
```

} **for each_list in app_data_set:**
 print(each_list)

Output

```
['Facebook', 0.0, 'USD', 2974676, 3.5]
['Instagram', 0.0, 'USD', 2161558, 4.5]
['Clash of Clans', 0.0, 'USD', 2130805, 4.5]
['Temple Run', 0.0, 'USD', 1724546, 4.5]
['Pandora - Music & Radio', 0.0, 'USD', 1126879, 4.0]
```

L'utilizzo della tecnica di cui sopra richiede la possibilità di scrivere una riga di codice per ogni riga nel set di dati. Ma l'utilizzo della metodologia `app_data_set` per ogni elenco implica che scriviamo solo due righe di codice indipendentemente dal numero di righe nel set di dati: il set di dati può avere cinque righe o centomila.

Il nostro obiettivo di transizione è utilizzare questo metodo speciale per calcolare la valutazione media delle nostre cinque righe sopra, in cui il nostro obiettivo finale è calcolare la valutazione media di 7.197 righe per il nostro set di dati. Otterremo esattamente questo nelle prossime dimostrazioni di questo compito, ma per ora ci concentreremo sulla pratica di questo metodo per comprenderlo bene.

Dovremmo far rientrare i caratteri dello spazio quattro volte a destra prima di voler scrivere il codice:

script.py

```
app_data_set = [row_1, row_2, row_3, row_4, row_5]
```

```
for each_list in app_data_set:  
    print(each_list)
```

**We need to indent the code we want repeated
four space characters to the right**

In teoria, dovremmo solo indentare il codice a destra con almeno un carattere spazio, ma nel linguaggio Python la dichiarazione prevede l'uso di quattro caratteri spazio. Questo aiuta la leggibilità

— leggere il tuo codice sarà abbastanza facile per le altre persone che guardano questa convention, e tu lo farai trovare più facile seguire i loro.

Ora usa questa tecnica per stampare il nome e la valutazione di ogni

```
app: foreach_list in app_data_set:
```

```
    nome = each_list[0]
```

```
    valutazione = ogni_lista[-1]
```

```
    stampa (nome, valutazione)
```

Facebook 3.5

Instagram 4.5

Clash of Clans 4.5

Temple Run 4.5

Pandora - Musica e radio 4.0

loop

Un ciclo viene spesso utilizzato per iterare su una serie di istruzioni. Abbiamo due tipi di loop, "for loop" e "while loop" in Python. Studieremo 'for loop' e 'while loop' nel seguente scenario.

Per il ciclo

Il ciclo for di Python viene utilizzato per iterare su una sequenza (elenco, tupla, stringa) o su qualsiasi oggetto in grado di iterare. Si chiama traversal per iterare su una sequenza.

Sintassi del ciclo For in Python

```
per<variabile> in <sequenza>:
```

```
    # body_of_loop che ha un insieme di istruzioni
```

```
    # che richiede un'esecuzione ripetuta
```

In questo caso < variabile > è spesso una variabile utilizzata per iterare su una < sequenza >. Intorno a ogni iterazione il valore successivo viene preso da < sequenza > per avvicinarsi alla fine della sequenza.

Python – Esempio di ciclo For

L'esempio seguente illustra l'uso di un ciclo per eseguire l'iterazione su un array di elenchi. Calcoliamo il quadrato di ogni numero presente nell'elenco e mostriamo lo stesso con il corpo del ciclo for.

```
# Programma di stampa quadrati di tutti i numeri
```

```
# Elenco di numeri interi
```

```
numeri = [1, 2, 4, 6, 11, 20]
```

```
# variabile per memorizzare il quadrato temporaneo di ogni numero
```

```
sq = 0
```

```
# iterando sull'elenco dato
```

```
for val in numeri:
```

```
    # calcolo del quadrato di ogni numero
```

```
    sq = val * val
```

```
    # visualizzare i quadrati
```

```
    stampa (quadrato)
```

```
Produzione:
```

1
4
16
36
121
400

Per loop con blocco else

Escludendo Java, possiamo avere il ciclo collegato con un blocco 'else' opzionale in Python. Il blocco "else" viene eseguito solo dopo che tutte le iterazioni sono terminate dal ciclo. Vediamo un esempio:

Per valore nell'intervallo (5):

 stampa (valore)

altro:

 print("Il ciclo ha completato l'esecuzione")

Output:

0
1
2
3
4

Il ciclo ha completato l'esecuzione

Nota: il blocco else viene eseguito al termine del ciclo.

Ciclo For annidato in Python

Se è presente un ciclo all'interno di un altro ciclo for, verrà definito un ciclo for nidificato. Prendiamo un esempio di ciclo for annidato.

per num1 nell'intervallo (3):

 per num2 nell'intervallo (10, 14):

 print(num1, ",", num2)

Produzione:

0, 10
0, 11
0, 12
0, 13
1, 10
1, 11
1, 12
1, 13
2, 10
2, 11
2, 12

2 , 13

Mentre il ciclo

Mentre il ciclo viene anche utilizzato per iterare continuamente su un blocco di codice fino a quando un'istruzione specificata non restituisce false, abbiamo visto in molti cicli for in Python nell'ultima guida, che viene utilizzata per un intento simile. La più grande distinzione è che usiamo per il ciclo quando non siamo sicuri di quante volte il ciclo deve essere eseguito, ma dall'altra parte quando ci rendiamo conto esattamente quante volte dobbiamo eseguire il ciclo, abbiamo bisogno di un ciclo.

Sintassi del ciclo while

mentre si condiziona:

```
# body_of_while
```

Il corpo del while è una serie di istruzioni di Python che richiedono un'implementazione ripetitiva. Queste attestazioni vengono eseguite in modo coerente finché la condizione specificata non restituisce false.

mentre il flusso del ciclo

1. Per prima cosa viene ispezionata la condizione data, il ciclo viene annullato se la condizione restituisce false, e anche il controllo si sposta verso l'istruzione successiva nel compilatore dopo il ciclo.

2. Quando la condizione restituisce true, verrà eseguita la serie di istruzioni all'interno del ciclo e power passerà quindi all'avvio del ciclo per l'esecuzione successiva.

Queste due misure si verificano continuamente finché la condizione definita nel ciclo rimane vera.

Esempio di ciclo while

Questo è un esempio di ciclo while. Abbiamo un numero variabile in questo caso e mostriamo il valore del numero in un ciclo, il ciclo avrà un'operazione incrementale in cui aumentiamo il valore del numero. È un componente molto cruciale, mentre il ciclo dovrebbe avere un'operazione di aumento o diminuzione. In caso contrario, il ciclo funzionerà a tempo indeterminato.

```
numero = 1
# loop si ripeterà finché può
# num< 10 rimane vero
mentre num< 10:
    stampa (numero)
    # incrementando il valore di num
    numero = numero + 3
```

Produzione:

1
4
7

Ciclo while infinito

Esempio 1:

Questo stamperà all'infinito la parola "ciao" poiché questa situazione sarà sempre vera. mentre vero:

```
print("ciao")
```

Esempio 2:

```
numero = 1
while numero < 5:
    stampa (numero)
```

Questo stamperà '1' all'infinito poiché non aggiorniamo il valore numerico all'interno del ciclo, quindi il valore numerico rimarrebbe sempre uno e la condizione `numero < 5` restituirebbe sempre `true`.

Ciclo while annidato in Python

Mentre all'interno di un altro ciclo `while` è presente un ciclo `while`, verrà considerato ciclo `while` annidato. Per comprendere questo concetto, facciamo un esempio.

```
io = 1
j = 5
mentre io < 4:
    mentre j < 8:
        print(i, ", ", j)
        j = j + 1
    io = io + 1
```

Produzione:

```
1 , 5
2, 6
3, 7
```

Python – ciclo while con blocco else

Possiamo aggiungere un blocco `"else"` a un ciclo `while`. La sezione `'altro'` è possibile. Viene eseguito solo quando l'elaborazione del ciclo è terminata.

```
numero = 10
mentre numero > 6:
    stampa (numero)
    numero = numero - 1
    altro:
        print("il ciclo è terminato")
```

Output:

```
10
9
8
7
Il ciclo è finito
```

AGGIUNTA DI PI DATI CON VALORE IN PITONE



Spesso il creatore desidera che gli utenti inseriscano più valori o input in una riga. In Python, gli utenti possono utilizzare due tecniche per prendere più valori o input in una riga.

oh

1. Uso del metodo `split()`
2. Uso della comprensione delle liste

Uso del metodo `split()`:

Questa funzione aiuta a ricevere molti input dell'utente. Divide il separatore definito nell'input dato. Se non viene fornito alcun separatore, un separatore è uno spazio vuoto. Gli utenti generalmente utilizzano un metodo `split()` per separare una stringa Python, ma può essere utilizzato quando vengono presi più input.

Sintassi:

```
input().split(separator, maxsplit)
```

Esempio:

```
filter_none
edit
play_arrow
brightness_4
#Python program showing how to add
#multiple input using split
#taking two inputs each time
x, y = input("Enter a two value: ").split()
print("Number of boys: ", x)
print("Number of girls: ", y)
print()
# taking three inputs at a time
x, y, z = input("Enter a three value: ").split()
print("Total number of students: ", x)
print("Number of boys is : ", y)
print("Number of girls is : ", z)
print()
# taking two inputs at a time
a, b = input("Enter a two value: ").split()
print("First number is {} and second number is {}".format(a, b))
print()
# taking multiple inputs at a time
# and type casting using list() function
x = list(map(int, input("Enter a multiple value: ").split()))
print("List of students: ", x)
```

Produzione:

```
Enter a two value: 5 10
Number of boys: 5
Number of girls: 10

Enter a three value: 30 10 20
Total number of students: 30
Number of boys is : 10
Number of girls is : 20

Enter a four value: 20 30
First number is 20 and second number is 30

Enter a multiple value: 20 30 10 22 23 26
List of students: [20, 30, 10, 22, 23, 26]
```

Utilizzo della comprensione delle liste:

La comprensione delle liste è un modo semplice per descrivere e costruire una lista in Python. Proprio come le dichiarazioni matematiche, possiamo generare elenchi solo all'interno di ogni riga. Viene spesso utilizzato quando si raccolgono più input di dispositivi.

Esempio:


```

filter_none
edit
play_arrow
brightness_4
# Python program showing
# how to take multiple input
# using List comprehension
# taking two input at a time
x, y = [int(x) for x in input("Enter two value: ").split()]
print("First Number is: ", x)
print("Second Number is: ", y)
print()
# taking three input at a time
x, y, z = [int(x) for x in input("Enter three value: ").split()]
print("First Number is: ", x)
print("Second Number is: ", y)
print("Third Number is: ", z)
print()
# taking two inputs at a time
x, y = [int(x) for x in input("Enter two value: ").split()]
print("First number is {} and second number is {}".format(x, y))
print()
# taking multiple inputs at a time
x = [int(x) for x in input("Enter multiple value: ").split()]
print("Number of list is: ", x)

```

Produzione:

```

Enter two value: 2 5
First Number is: 2
Second Number is: 5

Enter three value: 2 4 5
First Number is: 2
Second Number is: 4
Third Number is: 5

Enter two value: 2 10
First number is 2 and second number is 10

Enter multiple value: 1 2 3 4 5
Number of list is: [1, 2, 3, 4, 5]

```

Nota: le definizioni di cui sopra prendono input divisi per spazi. Se preferiamo perseguire un input diverso con la virgola (","), possiamo semplicemente usare quanto segue:

```
# prendendo più input divisi per virgola alla volta
x = [int(x) for x in input("Inserisci un valore multiplo: ").split(",")]
print("Il numero della lista è: ", x)
```

Assegna più valori a più variabili

Separando le variabili ei valori con virgole, è possibile allocare più valori a variabili diverse.

```
a, b = 100, 200
```

```
stampa(a)
```

```
# 100
```

```
stampa(b)
```

```
# 200
```

Hai più di tre variabili da delegare. Inoltre, possono essere assegnati anche vari tipi.

```
a, b, c = 0.1, 100, 'stringa'
```

```
print(a)
```

```
# 0.1
```

```
stampa(b)
```

```
# 100
```

```
stampa(c)
```

```
# corda
```

Assegna lo stesso valore a più variabili

Usando = consecutivamente, potresti anche nominare più variabili con lo stesso valore. Ad esempio, questo è utile quando si inizializzano più variabili quasi con lo stesso valore.

```
a = b = 100
```

```
stampa(a)
```

```
# 100
```

```
stampa(b)
```

```
# 100
```

Dopo aver definito lo stesso valore, anche un altro valore può essere convertito in uno. Come spiegato in seguito, quando si allocano oggetti mutevoli come elenchi o dizionari, occorre prestare attenzione.

```
a = 200
```

```
stampa(a)
```

```
# 200
```

```
stampa(b)
```

```
# 100
```

Se ne possono scrivere tre o più allo stesso modo. a =

```
b = c = 'stringa'
```

```
stampa(a)
```

```
# corda
```

```
stampa(b)
```

```
# corda
```

```
stampa(c)
```

```
# corda
```

Invece di oggetti immutabili come int, float e str, fare attenzione quando si nominano oggetti mutabili come list e dict.

Quando si usa = consecutivamente, a tutte le variabili viene assegnato lo stesso oggetto, quindi se si modifica il valore dell'elemento o si crea un nuovo elemento, anche l'altro oggetto verrà modificato.

```
a = b = [0, 1, 2]
```

```
print(a è b)
```

```
# Vero
```

```
a[0] = 100
```

```
stampa(a)
```

```
# [100, 1, 2]
```

```
stampa(b)
```

```
# [100, 1, 2]
```

Come sotto.

```
b = [0, 1, 2]
```

```
a = b
```

```
stampa (a è b)
```

```
# Vero
```

```
a[0] = 100
```

```
stampa(a)
```

```
# [100, 1, 2]
```

```
stampa(b)
```

```
# [100, 1, 2]
```

Se vuoi gestirli in autonomia devi allocarli separatamente.

dopo c = []; d = [], c e d sono garantiti per il collegamento a due elenchi vuoti e diversi univoci, appena creati.

(Nota che c = d = [] assegna lo stesso oggetto sia a c che a d.)

Ecco un altro esempio:

```
a = [0, 1, 2]
```

```
b = [0, 1, 2]
```

```
stampa (a è b)
```

```
# Falso
```

```
a[0] = 100
```

```
stampa(a)
```

```
# [100, 1, 2]
```

```
stampa(b)
```

[0, 1, 2]

AGGIUNGERE I DATI DI STRINGA IN PYTHON

Che cos'è String in Python?

UNA stringa è un set di caratteri. Un personaggio è solo un simbolo. La lingua inglese, ad esempio, ha 26 caratteri. I sistemi operativi non gestiscono i caratteri, ma gestiscono i numeri (binari). E se puoi vedere caratteri sul tuo computer, è rappresentato internamente come una combinazione di 0 e 1 e viene manipolato. La trasformazione del carattere in un numero è nota come codifica e probabilmente la decodifica è il processo inverso. ASCII e Unicode sono due delle codifiche ampiamente utilizzate. Una stringa in Python è una serie di caratteri in Unicode. Unicode è stato incorporato per fornire tutti i caratteri in tutte le lingue e per garantire l'uniformità di codifica. Python Unicode ti permette di imparare riguardo Unicode.

Come creare una stringa in Python?

Le stringhe possono essere formate incapsulando caratteri o anche virgolette doppie all'interno di una singola citazione. In Python, possono essere usate anche le virgolette triple, ma comunemente usate per rappresentare stringhe multilinea e docstring.

```
# defining strings in Python
```

```
# all of the following are equivalent
```

```
my_string = 'Hello'
```

```
print(my_string)
```

```
my_string = "Hello"
```

```
print(my_string)
```

```
my_string = '''Hello'''
```

```
print(my_string)
```

```
# triple quotes string can extend multiple lines
```

```
my_string = """Hello, welcome to the world of Python"""
```

```
print(my_string)
```

Quando il programma viene eseguito, l'output diventa:

```
Hello
```

```
Hello
```

```
Hello
```

```
Hello, welcome to the world of Python
```

Accedere ai caratteri in una stringa?

Indicizzando e usando lo slicing, possiamo ottenere singoli caratteri e l'ambito dei caratteri. L'indice inizia da 0. Il tentativo di ottenere un carattere dall'intervallo dell'indice causerà l'aumento di un `IndexError`. L'indice deve essere integrale. Non possiamo usare float o altri tipi e questo porterà a `TypeError`. Python permette che le sue sequenze siano indicizzate negativamente. L'indice -1 corrisponde all'ultimo oggetto, - 2 al secondo oggetto, e così via. Usando l'operatore di affettatura '(due punti)', possiamo accedere a un intervallo di elementi all'interno di una stringa.

Accesso ai caratteri stringa Python:

```
str = 'programiz'

print('str = ', str)

#first character

print('str[0] = ', str[0])

#last character

print('str[-1] = ', str[-1])

#slicing 2nd to 5th character

print('str[1:5] = ', str[1:5])

#slicing 6th to 2nd last character

print('str[5:-2] = ', str[5:-2])
```

If we execute the code above we have the following results:

```
str = programiz

str[0] = p

str[-1] = z

str[1:5] = rogr

str[5:-2] = am
```

Quando si tenta di accedere a un indice al di fuori dell'intervallo, o se si utilizzano numeri diversi da un numero intero, si verificano errori.

```
# indice deve essere compreso nell'intervallo
```

```
>>>my_string[15]
```

```
...
```

```
IndexError: string index out of range
```

```
# index must be an integer
```

```
>>>my_string[1.5]
```

```
...
```

TypeError: definisce gli indici di stringa solo come numeri interi

Analizzando l'indice tra gli elementi come mostrato di seguito, è possibile visualizzare al meglio lo slicing. Ogni volta che vogliamo ottenere un intervallo, abbiamo bisogno dell'indice che seziona la parte della stringa da esso.

Come modificare o eliminare una stringa?

Le stringhe sono immutabili. Ciò significa che gli elementi di un elenco non possono essere modificati finché non vengono assegnati. Riassegneremo facilmente varie stringhe dello stesso termine.

```
>>>my_string = 'programiz'
```

```
>>>my_string[5] = 'a'
```

```
...
```

```
TypeError: 'str' object does not support item assignment
```

```
>>>my_string = 'Python'
```

```
>>>my_string
```

```
'Python'
```

Non possiamo cancellare caratteri da una stringa o rimuoverli. Ma è facile cancellare completamente la stringa usando la parola chiave del.


```
>>>delmy_string[1]
```

```
...
```

```
TypeError: 'str' object doesn't support item deletion
```

```
>>>delmy_string
```

```
>>>my_string
```

```
...
```

```
NameError: name 'my_string' is not defined
```

Operazioni sulle stringhe Python

Ci sono molti metodi che possono essere usati con le stringhe che lo rendono uno dei tipi di dati Python più comunemente usati. Vedi Tipi di dati Python per maggiori informazioni sui tipi di dati usati nella codifica Python

Concatenazione di due o più stringhe

La combinazione di due o anche più stringhe in una è chiamata concatenazione. In Python, l'operatore + lo fa. Allo stesso modo vengono concatenati digitando insieme due stringhe letterali. Per un numero specificato di volte, è possibile utilizzare l'operatore * per reiterare la stringa.

```
# Python String Operations
```

```
str1 = 'Hello'
```

```
str2 = 'World!'
```

```
# using +
```

```
print('str1 + str2 = ', str1 + str2)
```

```
# using *
```

```
print('str1 * 3 = ', str1 * 3)
```

Once we execute the program above we get the following results:

```
str1 + str2 = HelloWorld!
```

```
str1 * 3 = HelloHelloHello
```

Using two literal strings together would therefore concatenate them like + operator.

We might use parentheses if we wish to concatenate strings in various lines.

```
>>> # two string literals together
```

```
>>> 'Hello ' 'World!'
```

```
'Hello World!'
```

```
>>> # using parentheses
```

```
>>> s = ('Hello '
```

```
...     'World')
```

```
>>>s
```

```
'Hello World'
```

Iterazione attraverso una stringa

Con un ciclo for, possiamo scorrere una stringa. Questo è un esempio di conteggio del numero di "l" in una funzione stringa.

```
#Iterating through a string

count = 0

for letter in 'Hello World':

    if(letter == 'l'):

        count += 1

print(count,'letters found')
```

Se eseguiamo il codice sopra, abbiamo i seguenti risultati: '3 lettere trovate.'

Test di appartenenza alla stringa

Possiamo verificare se c'è o meno una sottostringa all'interno di una stringa usando la parola chiave in.

```
>>> 'a' in 'programma'
```

Vero

```
>>> 'a' non in 'battaglia'
```

falso

Funzioni integrate per lavorare con Python

Diverse funzioni integrate che possono funzionare anche con stringhe in serie. Alcuni altri tipi comunemente usati sono len() ed enumerate(). La funzione enumerate() restituisce un oggetto enumerate. Include l'indice e il valore come combinazioni di tutti gli elementi nella stringa. Questo può essere utile per l'iterazione. In modo comparabile, len() restituisce la lunghezza della stringa (numero di caratteri).

```
str = 'cold'

# enumerate()

list_enumerate = list(enumerate(str))

print('list(enumerate(str) = ', list_enumerate)

#character count

print('len(str) = ', len(str))
```

Once we execute the code above we have the following results:

```
list(enumerate(str) = [(0, 'c'), (1, 'o'), (2, 'l'), (3, 'd')]

len(str) = 4
```

Formati per Python String

Sequence per l'escape

Non possiamo usare virgolette singole o doppie se vogliamo stampare un testo come disse: "Cosa c'è?" Ciò comporterebbe un errore di sintassi perché ci sono virgolette singole e doppie solo nel testo.

```
> > > print("Ha detto, "Cosa c'è?")
```

```
...
```

```
Errore di sintassi: sintassi non valida
```

```
> > > print('Ha detto, "Cosa c'è?")
```

```
...
```

```
Errore di sintassi: sintassi non valida
```

Le virgolette triple sono un modo per aggirare il problema. Potremmo usare le sequenze di escape come soluzione. Una serie di fughe inizia con una barra rovesciata, che è rappresentata in modo diverso. Se stiamo usando una singola virgoletta per descrivere una stringa, è importante evitare tutte le virgolette singole all'interno della stringa. Il caso delle virgolette è strettamente correlato. Ecco come si può rappresentare il testo sopra.

```
# using triple quotes

print(''He said, "What's there?''')

# escaping single quotes

print('He said, "What\'s there?')

# escaping double quotes

print("He said, \"What's there?\")
```

Una volta eseguito il codice sopra, abbiamo i seguenti risultati:

Ha detto "Cosa c'è?"

Disse: "Cosa c'è?"

Disse: "Cosa c'è?"

Stringa non elaborata per ignorare la sequenza di escape

Molto spesso all'interno di una stringa, potremmo voler rifiutare le sequenze di escape. Per usarlo, possiamo impostare `r` o `R` prima della stringa. Il che significa che è una stringa grezza e trascurerà qualsiasi sequenza di escape all'interno.

```
> > print("Questo è \x61 \nbuon esempio")
```

Questo è un

buon esempio

```
> > > print(r"Questo è \x61 \nbuon esempio")
```

Questo è \x61 \nbuon esempio

Il metodo `format()` per formattare le stringhe

I sorgenti `format()` disponibili e `make with the string object` sono molto flessibili e potenti nella formattazione delle stringhe. Le stringhe di stile contengono parentesi graffe `{}` come segnaposto o campi di sostituzione, che vengono sostituiti.

Per specificare la sequenza, possiamo usare argomenti posizionali o argomenti di parole chiave.

```

# Python string format() method
# default(implicit) order
default_order = "{}, {} and {}".format('John','Bill','Sean')
print('\n--- Default Order ---')
print(default_order)
# order using positional argument
positional_order = "{1}, {0} and {2}".format('John','Bill','Sean')
print('\n--- Positional Order ---')
print(positional_order)
# order using keyword argument
keyword_order = "{s}, {b} and {j}".format(j='John',b='Bill',s='Sean')
print('\n--- Keyword Order ---')
print(keyword_order)
Once we execute the code above we have the following results:
--- Default Order ---
John, Bill and Sean
--- Positional Order ---
Bill, John and Sean
--- Keyword Order ---
Sean, Bill and John

```

La tecnica `format()` può avere requisiti in formato facoltativo. Usando i due punti, vengono divisi dal nome del campo. Ad esempio, una stringa nello spazio dato può essere giustificata a sinistra `<`, giustificata a destra `>` o basata su `^`.

Anche noi possiamo formattare gli interi come binari, esadecimali, ecc. E i float possono essere arrotondati o mostrati nello stile dell'esponente. Puoi usare tonnellate di compilazione lì. Per tutta la formattazione delle stringhe disponibile utilizzando il metodo `format()`, vedere l'esempio seguente:

```

>>> # formatting integers

>>> "Binary representation of {0} is {0:b}".format(12)

'Binary representation of 12 is 1100'

>>> # formatting floats

>>> "Exponent representation: {0:e}".format(1566.345)

'Exponent representation: 1.566345e+03'

>>> # round off

>>> "One third is: {0:.3f}".format(1/3)

'One third is: 0.333'

>>> # string alignment

>>> "|{:<10}|{: ^10}|{:>10}|".format('butter', 'bread', 'ham')

'|butter    |  bread   |      ham|'

```

Formattazione vecchio stile

Possiamo persino codificare stringhe come il vecchio stile `sprint()` nel linguaggio di programmazione utilizzato in C. Per ottenere ciò; usiamo l'operatore '%'.

```

>>> x = 12.3456789

>>> print('The value of x is %3.2f' %x)

The value of x is 12.35

>>> print('The value of x is %3.4f' %x)

The value of x is 12.3457

```

Metodi comuni di stringhe per Python

L'oggetto stringa viene fornito con vari metodi. Uno di questi è il metodo `format()` che abbiamo descritto sopra. Alcune altre tecniche usate di frequente includono `lower()`, `upper()`, `join()`, `split()`, `find()`, `replace()` ecc. Ecco un ampio elenco di molte delle metodologie integrate in Python per lavorare con le stringhe.

```
>>> "PrOgRaMiZ".lower()

'programiz'

>>> "PrOgRaMiZ".upper()

'PROGRAMIZ'

>>> "This will split all words into a list".split()

['This', 'will', 'split', 'all', 'words', 'into', 'a', 'list']

>>> ' '.join(['This', 'will', 'join', 'all', 'words', 'into', 'a', 'string'])

'This will join all words into a string'

>>> 'Happy New Year'.find('ew')

7

>>> 'Happy New Year'.replace('Happy', 'Brilliant')

'Brilliant New Year'
```

Inserimento di valori nelle stringhe Metodo 1 -

il metodo del formato stringa

Il metodo di formattazione del metodo string può essere utilizzato per creare nuove stringhe con i valori inseriti. Questo metodo funziona per tutte le versioni recenti di Python. È qui che mettiamo una stringa in un'altra stringa:

```
>>>shepherd = "Mary"

>>>string_in_string = "Shepherd {} is on duty.".format(shepherd)

>>>print(string_in_string)
```

Il pastore Maria è di turno.

Le parentesi curve indicano dove andrà il valore inserito.

Puoi inserire un valore maggiore di uno. I valori non dovrebbero essere stringhe; numeri e altre entità Python possono essere stringhe.


```
>>>shepherd = "Mary"

>>>age = 32

>>>stuff_in_string = "Shepherd {} is {} years old.".format(shepherd, age)

>>>print(stuff_in_string)
```

Shepherd Mary is 32 years old.

```
>>> 'Here is a {} floating point number'.format(3.33333)

'Here is a 3.33333 floating point number'
```

Utilizzando le opzioni di formattazione all'interno delle parentesi graffe, puoi eseguire una formattazione più complessa di numeri e stringhe: consulta le informazioni sul layout delle stringhe tra parentesi graffe.

Questo processo ci consente di fornire istruzioni per la formattazione di elementi come i numeri, utilizzando: all'interno delle parentesi graffe, guidato dalla guida per la formattazione. Qui ti chiediamo di stampare in intero (d) in cui il numero è 0 per coprire la dimensione del campo di 3:

```
>>>print("Number {:03d} is here.".format(11))
```

Number 011 is here.

This prints a floating point value (f) with exactly 4 digits after the decimal point:

```
>>> 'A formatted number - {:.4f}'.format(.2)
```

'A formatted number - 0.2000'

Metodo 2 - f-stringhe in Python >= 3.6

Quando puoi contare sull'avere Python > = versione 3.6, avrai un altro posto interessante per usare la nuova stringa formattata letterale (f-string) per inserire i valori delle variabili. Proprio all'inizio della stringa, una f informa Python di consentire qualsiasi nome di variabile attualmente valido all'interno della stringa come nomi di colonna. Quindi ecco un esempio come quello sopra, ad esempio usando la sintassi f-string:

```
>>>shepherd = "Martha"

>>>age = 34

>>> # Note f before first quote of string

>>>stuff_in_string = f"Shepherd {shepherd} is {age} years old."

>>>print(stuff_in_string)
```

Il pastore Martha ha 34 anni.

Metodo 3 - formattazione % vecchia scuola

Sembra che ci sia un vecchio strumento di formattazione delle stringhe, che utilizza l'operatore percentuale. È un po' meno versatile delle altre due scelte, ma puoi ancora vederlo in uso nella codifica precedente, dove è più semplice usare la formattazione '%'. Per formattare l'operatore '%', dimostri dove dovrebbero andare i valori codificati usando un carattere '%' preceduto da un identificatore di formato per indicare come aggiungere il valore.

Quindi ecco l'esempio precedente in questo thread, utilizzando la formattazione di '%'. Nota che il marcatore '%s' per una stringa da inserire e il marcatore '%d' per un numero intero.

```
>>>stuff_in_string = "Shepherd %s is %d years old." % (shepherd, age)
```

```
>>>print(stuff_in_string)
```

Il pastore Martha ha 34 anni

Anni.

DATI DEL MODULO

Quali sono i moduli in Python?

W ogni volta che esci e rientri nell'interprete Python, le definizioni che hai creato (funzioni e variabili) andranno perse. Di conseguenza, se desideri sviluppare un codice un po' più a lungo, è meglio utilizzare un editor di testo per pianificare l'input per l'interprete ed eseguirlo con quel file come input al contrario. Questo è definito come formazione di script. Man mano che il software diventa più grande, potresti volerlo suddividere in file diversi per semplificare la manutenzione. Potresti anche usare una comoda funzione che hai scritto in molti altri programmi senza dover replicare la sua definizione all'interno di ogni programma. Per facilitare ciò, Python ha la possibilità di inserire definizioni in un file e di usarle nel codice dell'interprete o nelle istanze interattive. Questo stesso file è considerato un modulo;

Un modulo è un file che contiene definizioni e istruzioni da Python. Il nome del file è il nome del modulo con il suffisso .py allegato. Il nome del modulo (solo come stringa) all'interno di un modulo è disponibile come valore, inclusa la sua variabile globale `__name__`. Ad esempio, usa il tuo editor di testo preferito per creare un file chiamato `fibonacci.py` con i seguenti contenuti nella directory di lavoro corrente:

```
# Python Module example

def add(a, b):

    """This program adds two
    numbers and return the result"""

    result = a + b

    return result
```

In questo, abbiamo definito una funzione `add()` all'interno di un esempio intitolato "modulo". La funzione richiede due numeri e ne restituisce il totale.

Come importare i moduli in Python?

All'interno di un modulo, possiamo importare le definizioni in qualche altro modulo o anche nell'interprete Python interattivo. Per fare qualcosa del genere, usiamo la parola chiave `import`. Per caricare il nostro modulo di esempio specificato di recente, inserisci nel prompt di Python.

```
>>> esempio di importazione
```

Questo non dovrebbe importare le identità delle funzioni direttamente nella tabella dei simboli esistente, come definito nell'esempio. Importa solo un esempio del nome del modulo lì.

Usando il nome del modulo, possiamo usare l'operatore punto(.) per accedere alla funzione. Ad esempio:

```
> > > esempio.add(4,5.5)
```

```
9,5
```

Python viene fornito con molti moduli regolari. Dai un'occhiata all'elenco completo dei moduli Python regolari e ai loro scenari di utilizzo. Queste directory si trovano nella destinazione in cui hai installato Python nella directory Lib. I moduli normali possono essere importati allo stesso modo in cui vengono importati i nostri moduli definiti dall'utente.

Esistono diversi modi per importare i moduli. Li trovi di seguito:

Istruzione di importazione Python

Utilizzando l'istruzione import, possiamo estrarre un modulo utilizzando l'operatore punto, come spiegato nella sezione precedente e accedere alle definizioni al suo interno. Ecco un altro esempio.

```
# import statement example
```

```
# to import standard module math
```

```
import math
```

```
print("The value of pi is", math.pi)
```

Una volta eseguito il codice sopra, abbiamo i seguenti risultati: Il valore di pi è 3,141592653589793

Importa con rinomina

Possiamo caricare un modulo nel seguente modo cambiandone il nome:

```
# import module by renaming it
```

```
import math as m
```

```
print("The value of pi is", m.pi)
```

Abbiamo chiamato il modulo Math come m. In alcuni casi, questo ci farà risparmiare tempo per digitare. Ricorda che nel nostro ambito, il nome math non è identificato. Quindi math.pi non è corretto e m.pi è implementato correttamente.

Python da... dichiarazione di importazione

Possiamo importare singoli nomi da un tale modulo senza dover importare l'intero modulo. Ecco un altro esempio.

```
# import only pi from math module

from math import pi

print("The value of pi is", pi)
```

In questo, solo il parametro pi è stato importato dal modulo matematico. In alcuni casi non utilizziamo l'operatore punto. Allo stesso modo possiamo importare diversi moduli:

```
>>> da matematica import pi, e
>>> pi
3.141592653589933
>>> e
2.718281828459045
```

Importa tutti i nomi

Con il seguente modulo possiamo importare tutti i termini (definizioni) da un modulo:

```
# importa tutti i nomi dal modulo standard math
from math import *
print("Il valore di pi greco è," pi)
```

Sopra, abbiamo aggiunto tutte le descrizioni dei moduli matematici. Questo copre tutti i nomi disponibili nel nostro ambito tranne quelli che iniziano con un carattere di sottolineatura. Non è una buona tecnica di programmazione importare qualcosa con il tasto asterisco (*). Questo porterà a una replica del significato di un attributo. Questo limita anche la leggibilità del nostro codice.

Percorso di ricerca del modulo Python

Python esamina molte posizioni durante l'importazione di un modulo. L'interprete cerca invece un modulo integrato. Quindi, se non è incluso nel modulo integrato, Python cerca in una raccolta di directory specificata in sys.path. L'esplorazione è in questa sequenza:

PYTHONPATH (elenco delle variabili di ambiente delle directory) La
directory predefinita dipendente dall'installazione

```
>>> import sys

>>> sys.path

['',
 'C:\\Python33\\Lib\\idlelib',
 'C:\\Windows\\system32\\python33.zip',
 'C:\\Python33\\DLLs',
 'C:\\Python33\\lib',
 'C:\\Python33',
 'C:\\Python33\\lib\\site-packages']
```

Possiamo inserire quell'elenco e personalizzarlo per inserire la nostra posizione.

Ricaricare un modulo

Durante una sessione, l'interprete Python deve importare un modulo solo una volta. Questo rende le cose più produttive. Ecco un esempio che mostra come funziona.

Supponiamo di ottenere il codice seguente in un modulo chiamato my_module:

```
# This module shows the effect of

# multiple imports and reload

print("This code got executed")
```

Ora sospettiamo che più importazioni abbiano un impatto.

```
>>> importa mio_modulo
```

Questo codice è stato eseguito:

```
>>> importa mio_modulo
```

```
>>> importa mio_modulo
```

Abbiamo visto che il nostro codice è stato eseguito solo una volta. Ciò significa che il nostro modulo è stato importato solo una volta.

Inoltre, se durante il processo di test il nostro modulo è stato modificato, dovremo riavviarlo. Il modo per farlo è ricaricare l'interprete. Ma questo non aiuta enormemente. Python offre un modo efficace per farlo.

All'interno del modulo `imp`, possiamo usare la funzione `reload()` per riavviare un modulo. Ecco alcuni modi per farlo:

```
> > > importa imp
```

```
> > > importa mio_modulo
```

Questo codice viene eseguito

```
> > > importa mio_modulo
```

```
> > > imp.reload(mio_modulo)
```

Questo codice viene eseguito

```
<modulo 'my_module' da '.\\my_module.py'>
```

La funzione integrata `dir()`

Possiamo usare la funzione `dir()` per individuare i nomi specificati all'interno di un modulo. Per tali casi, nell'esempio del modulo che avevamo nella prima parte, abbiamo descritto una funzione `add()`.

Nel modulo di esempio, possiamo usare `dir` nel seguente scenario:

```
>>>dir(example)
```

```
['__builtins__',
```

```
'__cached__',
```

```
'__doc__',
```

```
'__file__',
```

```
'__initializing__',
```

```
'__loader__',
```

```
'__name__',
```

```
'__package__',
```

```
'add']
```

Ora vedremo un elenco dei nomi ordinati (a fianco di `add`). Molti altri nomi che iniziano con un carattere di sottolineatura sono attributi Python predefiniti associati al modulo (non definiti dall'utente). Ad esempio, il nome dell'attributo contiene il modulo `__name__`.

```
> > > esempio di importazione
```

```
> > > esempio.__name__
```

'esempio'

Puoi scoprire tutti i nomi identificati nel nostro spazio dei nomi esistente usando la funzione `dir()` senza argomenti.

```
>>> a = 1
```

```
>>> b = "hello"
```

```
>>> import math
```

```
>>> dir()
```

```
['__name__', '__doc__', '__builtins__', 'a', 'b', 'math', 'pyscripter']
```

Esecuzione di moduli come script

Modulo Python in esecuzione con `python fibo.py <arguments>` il programma verrà eseguito in questo modo, proprio come veniva importato, ma includendo `__name__` impostato su `"__main__"`. Ciò implica che questo programma sia inserito alla fine del modulo:

```
If __name__ == "__main__": import sys fib(int(sys.argv[1]))
```

Potresti anche creare il file utilizzabile sia come script che come modulo importabile poiché questo codice che analizza l'interfaccia a riga di comando viene eseguito solo quando il modulo viene eseguito come file "principale":

```
$ python fibo.py 50
```

```
0 1 1 2 3 5 8 13
```

Quando il modulo viene importato, il codice non verrà eseguito:

```
>>>
```

```
>>> importa fibo
```

```
>>>
```

Viene spesso utilizzato per ottenere un'interfaccia utente efficiente per un modulo o per scopi di test (il modulo esegue una suite di test come script).

File Python "compilati"

Per velocizzare il caricamento dei moduli, Python memorizza nella cache la versione compilata di ciascun modulo nella directory `__pycache__` con il nome `module.version.pyc`, in cui la versione incapsula il formato del file assemblato; normalmente include la versione del firmware di Python. Ad esempio, l'edizione compilata di `spam.py` in CPython launch 3.3 verrà memorizzata nella cache come `__pycache__/spam.cpython-33.pyc`. Questa convenzione di denominazione consente la coesistenza di moduli compilati da vari aggiornamenti e versioni separate di Python.

Python verifica la pianificazione delle modifiche all'origine rispetto all'edizione compilata per vedere se è obsoleta e necessita di una ricompilazione. Questo è un sistema completamente automatizzato. Anche i moduli assemblati diventano indipendenti dalla piattaforma, quindi algoritmi diversi utilizzeranno la stessa libreria tra i sistemi. In due situazioni Python non controllerà la cache:

- Innanzitutto, ricompila spesso l'output per il modulo, che viene caricato esplicitamente dalla riga di comando ma non lo memorizza.
- In secondo luogo, quando non c'è un modulo root, non cercherà nella cache. Il modulo compilato deve trovarsi nella directory dei sorgenti per facilitare un rilascio non sorgente (solo compilato) e un modulo sorgente non deve essere installato.

Alcuni suggerimenti per gli utenti:

- Per ridurre al minimo le dimensioni di un file compilato, puoi utilizzare le opzioni -O o -OO nell'ordine Python. L'opzione -O cancella le istruzioni di assert, l'opzione -OO rimuove le istruzioni di assert e le stringhe di doc. Sebbene alcuni codici possano supportare la disponibilità di queste opzioni, questo metodo dovrebbe essere utilizzato solo se si è consapevoli di ciò che si sta facendo. I moduli "ottimizzati" di solito hanno un tag opt e sono più piccoli. Le versioni future potrebbero modificare le implicazioni del controllo ottimale.
- Un progetto non viene eseguito più velocemente una volta letto da un file.pyc rispetto a come è stato letto da un file.py; solo una cosa sui file .pyc che sono più veloci nella velocità con cui verranno caricati.
- Una compilazione di tutti i moduli può generare file .pyc in una directory per tutti gli altri moduli.
- Maggiori dettagli su questo processo sono forniti in PEP 3147, insieme a un diagramma di flusso del processo decisionale.

Moduli standard

Python ha una libreria di moduli standard, menzionata in una sezione separata, l'allusione alla libreria Python (di seguito "Library Reference"). Alcuni moduli sono incorporati nell'interprete; che forniscono un'esposizione diretta a processi che non sono componenti della base del linguaggio ma sono comunque incorporati, sia per l'efficacia che per fornire accesso a sistemi operativi primitivi come le chiamate del codice sorgente. La raccolta di questi moduli è un'alternativa da personalizzare e si affida anche al framework sottostante. Il modulo winreg, ad esempio, è disponibile solo su Microsoft Windows. Un modulo in particolare è degno di un certo interesse: sys, che è integrato in ogni interprete Python. Le variabili sys.ps1 e sys.ps2 classificano le stringhe utilizzate come istruzioni primarie e secondarie:

```
> > >
> > > importa sistema
> > > sys.ps1
'>>> '
> > > sys.ps2
'...'
> > > sys.ps1 = 'C> '
C>print('Che schifo!')
Che schifo!
C>
```

Solo quando l'interprete è in modalità interattiva queste due variabili vengono definite. La variabile sys.path è una raccolta di stringhe che definisce il percorso di ricerca per i moduli utilizzati dall'interprete. Quando PYTHONPATH non fa parte dell'insieme, verrà definito in un percorso predefinito preso dalla variabile di ambiente PYTHONPATH o tramite un'impostazione predefinita. Puoi cambiarlo con le normali procedure di elenco:

```
> > >
> > > importa sistema
> > > sys.path.append('/python/ufs/guido/lib/')

```

Pacchetti

I pacchetti sono infatti un modo per costruire lo spazio dei nomi del modulo Python usando "nomi puntati del modulo". Ad esempio, in un pacchetto chiamato A., il titolo del modulo AB specifica un sottomodulo chiamato B. Anche se l'uso dei moduli impedisce agli autori dei vari moduli di smettere di conoscere i nomi delle variabili globali l'uno dell'altro, qualsiasi uso del modulo puntato nomi impedisce agli sviluppatori di bundle multi-modulo come NumPy o Pillow di doversi preoccupare maggiormente dei nomi dei moduli l'uno dell'altro. Considera la possibilità di creare una serie di elenchi di moduli (un "pacchetto") per gestire file audio e dati audio in modo uniforme.

Esistono diversi programmi di file audio che di solito hanno familiarità con la loro estensione, ad esempio: "wav,.aiff,.au", anche se dovrai creare e mantenere un'enorme raccolta di moduli per convertire tra alcuni dei molteplici formati di File. Ci sono molte altre diverse operazioni che potresti voler eseguire sui dati audio (come la fusione, l'aggiunta di eco, l'implementazione di una funzione di equalizzazione, la produzione di un effetto stereo ottico), e dovrai solo scrivere una serie infinita di moduli per eseguirli interventi. Ecco un altro layout del pacchetto fattibile (descritto in termini di un file system gerarchico):

```

sound/                                Top level package
    __init__.py                       sound package initialization
formats/                             Subpackage for conversions of file format
    __init__.py
    wavread.py
    wavwrite.py
    aiffread.py
    aiffwrite.py
    auread.py
    auwrite.py
    ...
effects/                             Sound effectssubpackage
    __init__.py
    echo.py
    surround.py
    reverse.py
    ...
filters/                             Filterssubpackage
    __init__.py
    equalizer.py
    vocoder.py
    karaoke.py
    ...

```

Durante il caricamento del bundle, Python verifica la sottodirectory del pacchetto tramite le cartelle su sys.path. Per consentire le directory di visualizzazione Python che contengono il file come pacchetti, sono necessari i file __init__.py. Ciò protegge le directory con un nome comune, inclusa la stringa, dal nascondere accidentalmente moduli validi, che in seguito appaiono principalmente nel percorso di ricerca del modulo. Nell'ordine corretto; __init__.py può essere solo un file vuoto, ma potrebbe anche implementare il codice di pre-elaborazione del pacchetto o stabilire la variabile __all__ descritta di seguito

- Gli utenti del pacchetto possono anche caricare singoli moduli dal pacchetto, come: `'import sound.effects.echo'`
- Questo carica il sottomodulo `'sound.effects.echo'`. Il suo nome completo deve essere menzionato: `'sound.effects.echo.echofilter(input, output, atten=4, delay=0.7)'`
- Un altro modo per importare il sottomodulo è: `'fromsound.effects import echo'`
- Pertanto, avvia il sottomodulo echo e fornisce l'accesso ma senza il prefisso del pacchetto: `'echo.echofilter(input, output, atten=4, delay=0.7)'`
- E solo un'altra opzione è importare esplicitamente la funzione o l'attributo desiderato: `'fromsound.effects.echo import echofilter'`
- Questo attiva nuovamente il sottomodulo echo, tuttavia ciò abilita la sua funzione `echofilter()` esplicitamente accessibile: `'echofilter(input, output, delay=0.7, atten=4)'`

Quindi accumula l'eco del sottomodulo; tuttavia questo tende a fare la sua funzione; ricorda che l'oggetto sarà un sottomodulo (o sottopacchetto) del pacchetto o qualsiasi altro nome descritto nel pacchetto, come una funzione, una classe o una variabile durante l'utilizzo dell'oggetto di importazione dal pacchetto. Inizialmente, l'istruzione `import` analizza se l'oggetto è caratterizzato nel pacchetto; in caso contrario, suppone che sia un modulo e tenta di caricarlo. Se non riesce a raggiungerlo, verrà promossa un'eccezione a `"ImportError"`.

Facendo riferimento a questo, mentre si usa una sintassi come `import 'item.subitem.subsubitem'`, ogni articolo deve essere un pacchetto, ma l'ultimo; l'ultimo elemento potrebbe essere un modulo o un pacchetto, ma questo non può essere una classe o una funzione o una variabile identificata nell'elemento precedente.

CONCLUSIONE

La ricerca in quasi tutti i campi è diventata più orientata ai dati, influenzando sia le opportunità di lavoro che le competenze richieste. Mentre stanno diventando disponibili più dati e metodi per valutarli, stanno diventando più aspetti dipendenti dai dati dell'economia, della società e della vita quotidiana. Quando si tratta di data science, Python è uno strumento necessario con tutti i tipi di vantaggi. È flessibile e in continuo miglioramento perché è open-source. Python ha già una serie di preziose librerie e non si può ignorare che può essere combinato con altri linguaggi (come Java) e framework attuali. Per farla breve: Python è un metodo straordinario per la scienza dei dati.

"UN GRANDE BUSINESS INIZIA IN PICCOLO"

RICHARD BRANSON

COMMERCIO DI OPZIONI PER PRINCIPIANTI

Scambia per vivere e guadagna un reddito passivo extra. Investi da casa, impara a fare oscillare le azioni. Suggerimenti sulla gestione del rischio. Ottieni la libertà finanziaria con un ROI positivo

tra 7 giorni

MARK BROKER

© Copyright 2020 - Tutti i diritti riservati.

Il contenuto di questo libro non può essere riprodotto, duplicato o trasmesso senza il permesso scritto diretto dell'autore o dell'editore.

In nessun caso sarà ritenuta responsabile alcuna colpa o responsabilità legale nei confronti dell'editore, o dell'autore, per eventuali danni, riparazioni o perdite monetarie dovute alle informazioni contenute in questo libro. O direttamente o indirettamente.

Avviso legale:

Questo libro è protetto da copyright. Questo libro è solo per uso personale. Non è possibile modificare, distribuire, vendere, utilizzare, citare o parafrasare alcuna parte o il contenuto di questo libro senza il consenso dell'autore o dell'editore.

Avviso di esclusione di responsabilità:

Si prega di notare che le informazioni contenute in questo documento sono solo a scopo educativo e di intrattenimento. È stato compiuto ogni sforzo per presentare informazioni accurate, aggiornate, affidabili e complete. Nessuna garanzia di alcun tipo è dichiarata o implicita. I lettori riconoscono che l'autore non si impegna a fornire consulenza legale, finanziaria, medica o professionale. Il contenuto di questo libro è stato derivato da varie fonti. Si prega di consultare un professionista autorizzato prima di tentare qualsiasi tecnica descritta in questo libro.

Leggendo questo documento, il lettore accetta che in nessun caso l'autore è responsabile per eventuali perdite, dirette o indirette, sostenute a seguito dell'uso delle informazioni contenute in questo documento, inclusi, ma non limitati a, — errori, omissioni o imprecisioni.

INTRODUZIONE

Oh! Out di molte idee sbagliate, quella che circonda molti mercati è che le opzioni sono rischiose. Bene, se chiedi a un trader di opzioni, non accetterà questo. Il motivo è che se il trading di opzioni fosse rischioso, sarebbe stato un concetto obsoleto nel mercato. Perché sempre più trader e investitori si lanciano in questo business?

C'è solo la paura che fa pensare alla gente che le opzioni non siano redditizie. Tutto quello che devi fare è afferrare attentamente tutti i concetti e applicarli quando necessario. Inoltre, assicurati di scegliere la strategia giusta e al momento giusto.

"Investire dovrebbe essere più come guardare la vernice secca o guardare l'erba crescere. Se vuoi eccitazione, prendi \$ 800 e vai a Las Vegas". - Paul Samuelson

Abbiate un po' di pazienza poiché l'investimento non è uno scherzo! Secondo una stima del ballpark, un principiante ha bisogno di almeno uno o due anni per diventare un trader di grande successo.

Puoi fare affidamento su questa guida per principianti sul trading di opzioni che contiene tutti i concetti di base, i suggerimenti, le tecniche e le soluzioni relative al trading di opzioni. Leggi attentamente la teoria e poi implementa i concetti forniti in essa uno per uno. Ecco un autentico mantra per il trading di opzioni:

Ogni volta che fai una strategia di trading, pensaci almeno tre volte prima della tua chiamata finale: questa è la vera ricetta per il successo!

1. FONDAMENTI DI NEGOZIAZIONE DI OPZIONI



“Non investire mai in un business che non puoi capire”, dice Warren Buffett, il famoso investitore americano, e noi siamo d'accordo con lui! Ecco perché lo faremo inizia da zero per te in questo libro.

in questo capitolo imparerai le basi del trading di opzioni con esempi. Ti suggeriamo di prendere nota dei nuovi concetti che incontri qui in modo da poter assorbire più del previsto.

Ecco un suggerimento per professionisti: concentrati sui concetti e sui termini forniti in questo. Questo ti aiuterà ad afferrare le basi davvero bene. Allora, cominciamo?

Che cos'è il trading di opzioni?

All'inizio, il trading di opzioni sembra opprimente, ma è molto facile da capire se inizi da zero. Con questo intendiamo se parti da un concetto all'altro.

Fondamentalmente, i portafogli dei trader vengono creati con diverse categorie di asset. Ad esempio, possono essere ETF, azioni, fondi comuni di investimento e obbligazioni, ecc. Le opzioni sono una specie di categoria di attività con molti fronzoli. Ciò significa che possono darti più vantaggi rispetto agli ETF e alle azioni.

Usi delle Opzioni

Le opzioni possono rendere potente un trader. È perché possono aumentare il reddito, la leva finanziaria e la protezione di un trader. Per ogni investitore è presente uno scenario di opzione. Ad esempio, si possono utilizzare le opzioni come copertura vantaggiosa contro un mercato azionario in calo per limitare le perdite. Possono anche essere utilizzati per realizzare guadagni ricorrenti. Inoltre, le opzioni sono utilizzate anche per 'scopi speculativi' come scommettere sul movimento delle azioni, eccetera.

Con obbligazioni e azioni, non c'è niente come il pranzo gratis. Anche le opzioni non sono diverse. Comporta rischi e gli investitori dovrebbero esserne consapevoli.

Derivati

I derivati sono considerati come un gruppo "più grande" di titoli - e opzioni

appartenere a loro. Il prezzo di un derivato è "derivato" da qualcos'altro. Ad esempio, il ketchup è il derivato dei pomodori e le patatine fritte sono il derivato delle patate. Allo stesso modo, un'opzione su azioni è il derivato di azioni, mentre le opzioni sono derivate da titoli finanziari.

Alcuni degli esempi di derivati includono put, call, titoli garantiti da ipoteca, futures, swap, forward e altro.

Quindi, cosa intendi per opzioni?

- Le opzioni sono fundamentalmente contratti. Consentono agli acquirenti i diritti (non gli obblighi) di acquistare o vendere (in caso di call o put) un asset specifico a un determinato prezzo o prima della data di scadenza specificata.
- Gli investitori li utilizzano per generare reddito, coprire i rischi o speculare.
- Sono chiamati derivati. Il motivo è che le opzioni derivano il loro valore dalle attività sottostanti.
- Un contratto (di opzione su azioni) ha tipicamente 100 azioni di (un sottostante), ma possono essere scritte su qualsiasi tipo di attività sottostante da valute, obbligazioni a materie prime.

Caratteristiche del trading di opzioni

Tutte le opzioni scadono

Ricorda, tutte le opzioni scadono un giorno. Ciò significa che "MUORE" dopo il giorno di scadenza. Questa scadenza potrebbe essere dopo due giorni o due anni. Ciò significa che i trader devono pensare al tempo di scadenza prima di acquistare un'opzione.

Le scorte possono essere detenute per tutta la vita, d'altra parte.

Tutte le opzioni hanno un prezzo di esercizio

C'è un "prezzo di esercizio" per ogni opzione. Questo è il principio in cui un'opzione può essere convertita in "azioni di azioni". Ad esempio, se c'è un prezzo di esercizio fissato per un'opzione a \$ 109. È possibile utilizzare l'opzione per acquistare/vendere azioni a questo prezzo di esercizio.

Moltiplicatore del contratto di opzione

Supponiamo che ci sia una quota di azioni con un prezzo di \$ 105. Può essere acquistato a 105 dollari. Quando un'opzione è di \$ 6,00, NON PU essere acquistata a \$ 6,00. preferiresti aver bisogno di \$ 600 per comprarlo.

Il motivo è che le opzioni possono essere negoziate con 100 azioni. Ciò significa che è necessario moltiplicare il prezzo di un'opzione con 100 per raggiungere il suo "premio".

Tipi di opzioni

Il trading di opzioni ha un immenso potenziale al rialzo con un rischio limitato. Ci sono due tipi principali di Opzioni.

- Opzioni di chiamata

Il prezzo dell'opzione call mostra un movimento al rialzo quando il prezzo delle azioni aumenta e inizia a scendere quando il prezzo delle azioni scende. Significa che puoi dire che è direttamente proporzionale ai prezzi delle azioni.

Il prezzo dell'opzione call si sposta con il prezzo delle azioni!

Si possono condividere 100 azioni con il prezzo di esercizio di un'Opzione Call. Supponiamo che ci sia un appartamento in affitto e il suo prezzo sia di \$ 200.000. Vuoi acquistare questo appartamento, pensando che il suo valore sarà raddoppiato dopo qualche tempo. Tuttavia, non vuoi pagare l'intero prezzo di questo appartamento.

Cosa fare?

È possibile acquistare un'opzione di chiamata per questo appartamento. Questa opzione ti consentirà di effettuare questo acquisto (dell'importo di \$ 200.000) in 24 mesi. Ma questo processo comporterà un contratto e dovrai pagare per quel contratto.

Questo contratto finanziario è noto come "Opzione".

Quindi, il prezzo di esercizio di questa opzione sarà di \$ 200.000 con data di scadenza di 24 mesi. Il vantaggio di questo è che se il prezzo dell'appartamento aumenta durante questo periodo, non ti influenzerà (non dovrai pagare un extra su questo).

Ora, immaginiamo che accada il contrario. Il prezzo dell'appartamento non aumenta di valore. Piuttosto diminuisce dopo 24 mesi a \$ 150.000.

In questo caso, non sei obbligato ad acquistare l'appartamento perché hai la possibilità di non acquistarlo. Tenendo presente il prezzo ridotto di \$ 150.000, non sceglierai di acquistarlo al prezzo di esercizio di \$ 200.000.

Dal momento che hai pagato il contratto a un prezzo minimo (il contratto), perdi solo quello. Ora confronta questa perdita con l'opzione di acquistare la casa pagando l'intero prezzo in una volta. Avresti perso \$ 200.000 o (almeno \$ 50.000), vero?

Cos'è la chiamata? Cos'è Mettere?

Un'opzione call conferisce a un investitore il diritto di acquistare azioni, mentre l'opzione put gli consente di venderle. Ecco un esempio dell'opzione Call. Una persona potrebbe essere interessata all'acquisto di un nuovo appartamento in un nuovo edificio in costruzione vicino alla sua località. Tuttavia, vorrebbe pagarlo solo una volta completati i lavori di costruzione.

Quella persona può approfittare della sua opzione di acquisto. Attraverso l'opzione, può acquistarlo dal proprietario nei prossimi quattro anni a (diciamo) \$ 400.000 come acconto. Questo costo sarà chiamato "premium".

Ecco un esempio messo. Supponiamo di acquistare una casa e, con essa, di acquistare anche una polizza assicurativa per il proprietario di una casa. Questa polizza aiuta a proteggere la tua proprietà dai danni. Devi pagare un premio per questo per un periodo di tempo fisso. Questo premio è molto prezioso e aiuta a proteggere l'assicurato in caso di incidente domestico.

Supponiamo che, invece di un appartamento, il tuo bene sia un investimento indicizzato o azioni. Quindi, se un trader vuole acquistare un'assicurazione sul suo indice S&P 500, può acquistare opzioni put.

Supponiamo di nuovo di prevedere un mercato ribassista in futuro e di non voler perdere più del 10 o 11 per cento in quell'indice. Se l'Indice viene scambiato a \$2800 (per esempio), puoi acquistare un'opzione put, che ti renderà idoneo a vendere l'Indice a \$2550 in qualsiasi momento prima della data di scadenza.

Ciò contribuirà a ridurre la tua perdita. Anche se il mercato scende a zero, la tua perdita non sarebbe superiore al 10 per cento, nel caso in cui detieni l'opzione put.

Comprare e vendere

Le opzioni ti consentono di fare quattro cose:

- Vendi chiamate
- Vendi put
- Acquista chiamate
- Acquista Put

Tieni a mente questi quattro scenari perché questo è importante quando entri nel business del trading. L'acquisto di azioni offre una posizione lunga per gli investitori. Ma l'acquisto di un'opzione call può estendere la tua posizione (può renderla ancora più lunga). La vendita allo scoperto offre una posizione più breve. In un titolo sottostante, la vendita di una call scoperta dà anche una posizione corta.

Allo stesso modo, anche l'acquisto di put crea una posizione corta per te nel

titoli sottostanti. Mentre vendere put nude ti offre una posizione più lunga.

Ricorda che gli acquirenti di opzioni sono noti come titolari e i venditori di opzioni sono noti come "Scrittori" delle opzioni.

1. Non vi è alcun obbligo di acquisto o vendita per i detentori di call e put. Loro hanno i loro diritti. L'unico rischio per loro è quello di spendere soldi per l'acquisto di premium.

2. Tuttavia, è importante chiamare e mettere gli scrittori a comprare e vendere nel caso la loro opzione scade. Ciò significa che possono guadagnare di più, ma hanno anche un livello di rischio più elevato rispetto ai titolari.

2. PERCHÉ COMMERCIALE CON LE OPZIONI

Il trading di ptions è iniziato per la prima volta nel 1973. Possono offrire molti vantaggi ai singoli trader, sebbene abbiano la reputazione di essere rischiosi. Bene, devi pensare, quali sono questi vantaggi, vero? Ecco la risposta a questo.

I vantaggi del trading di opzioni

Sebbene le opzioni siano in circolazione da un po' di tempo ormai, la maggior parte degli investitori "teme" ancora di usarle. Il motivo è la minore informazione e l'uso scorretto. Ciò significa che se hai una buona conoscenza di tutte le basi delle opzioni (come stiamo fornendo qui), è più probabile che tu abbia successo come investitore.

I singoli investitori dovrebbero essere consapevoli dell'uso corretto e dei vantaggi delle opzioni prima di seguire ciecamente le voci secondo cui le opzioni sono "rischiose".

Le opzioni hanno un basso rischio

Alcune situazioni richiedono un rischio elevato per l'acquisto di opzioni rispetto al possesso di azioni. Tuttavia, ci sono anche scenari in cui l'utilizzo del trading di opzioni diventa la strategia migliore. Questo dipende anche da come li usi correttamente. Le opzioni richiedono un impegno finanziario basso rispetto alle azioni. Inoltre, sono impermeabili, il che promette meno rischi.

Un'altra qualità delle opzioni è confrontata con le azioni; le opzioni sono più sicure. Sono protetti dall'ordine di perdita di azioni. Questo ordine aiuta a fermare le perdite sotto un prezzo predeterminato indicato dal trader. Tuttavia, anche la natura dell'ordine è molto importante.

Supponiamo che un trader acquisti un titolo investendo \$50. Non vuole perdere più del 10 per cento; piazza un ordine di stop da \$45. Diventa un ordine di mercato quando il titolo viene scambiato al di sotto di questo prezzo. Questo ordine può funzionare durante il giorno, non durante la notte.

Ad esempio, le azioni chiudono a \$ 51, ma il giorno dopo si sentono cattive notizie sulle azioni come il proprietario dell'azienda mente sui guadagni o viene rilevata l'appropriazione indebita. Le azioni potrebbero aprire a 20 dollari. Se ciò accade, questo prezzo sarebbe la prima operazione al di sotto del prezzo dell'ordine di arresto dell'investitore. Il commerciante

venderebbe a questo prezzo (\$ 20), bloccando la perdita.

A sua tutela, se il commerciante avesse acquistato la put, non avrebbe subito tale perdita. Le opzioni non si chiudono quando il mercato scende e chiude. Questo accade con gli ordini di stop. Significato, gli ordini di stop si chiudono se il mercato si chiude.

Le opzioni mantengono i trader coperti 24 ore su 24, 7 giorni su 7. Gli ordini stop non possono fornire un'assicurazione 24 ore. Questo è il motivo per cui le opzioni sono considerate una "forma affidabile di copertura".

Le opzioni sono più convenienti

Con un maggiore potere di leva, le opzioni possono aiutarti a risparmiare un sacco di soldi. È possibile ottenere una posizione di opzione allo stesso modo in cui si ottiene una posizione di azioni. Per acquistare 200 azioni di un'azione del valore di \$ 80, devi pagare \$ 16.000, ad esempio. Ma se vuoi acquistare doppie chiamate del valore di \$ 20 (che rappresentano un contratto di 100 azioni), la spesa totale sarebbe di \$ 4000. Come?

Prova questa formula: 2 contratti si moltiplicano per 100 azioni divise per contratto x 20 \$ di prezzo di mercato. Avrai \$ 12.000 aggiuntivi da utilizzare a tua discrezione.

Anche se non è così semplice, richiede una buona comprensione e una buona strategia. Dovrai scegliere la chiamata giusta al momento giusto per acquistare per imitare la posizione del titolo nel modo corretto. Questa strategia è nota come sostituzione delle scorte, che non è solo praticabile ma anche conveniente e pratica.

Supponiamo che tu voglia acquistare Schlumberger (SLB, pensando che potrebbe aumentare di valore nei prossimi mesi. Pensi di dover acquistare 200 azioni e la società è scambiata a \$ 131. Quindi, il tuo overlay sarebbe \$ 26.200.

Invece di investire una somma così pesante, potresti scegliere opzioni per imitare le azioni e acquistare un'opzione call chiamata agosto, utilizzando solo un prezzo di esercizio di \$ 100 per \$ 34.

Se si desidera acquisire una posizione pari alle dimensioni di 200 azioni sopra menzionate, è necessario acquistare contratti doppi. Il tuo investimento totale per questo sarebbe \$ 6.800, invece di \$ 26.200. (Ecco come: doppi contatti x 100 azioni/contratto x prezzo di mercato di \$ 34). Puoi anche interessarti a questo o usare i tuoi soldi per un altro investimento.

Le opzioni offrono rendimenti più elevati

Il trading di opzioni promette rendimenti percentuali più elevati. I trader non hanno bisogno di una calcolatrice per trovarlo. Possono investire un importo basso e ottenere indietro un importo maggiore.

Consideriamo il caso d'uso sopra per confrontare il ritorno sull'investimento. I trader devono acquistare azioni per \$ 50 e un'opzione per \$ 6. Supponiamo che il prezzo delle opzioni cambi dell'80% (del prezzo delle azioni). Se le azioni salgono a \$ 5,5, un trader otterrà un rendimento del 10%. Ma l'opzione guadagnerebbe l'80 per cento del prezzo delle azioni di \$ 4,5. Un ritorno di questo tipo su un investimento di \$ 6 ammonta al 67,5%, che è molto meglio di un profitto del 10% sulle azioni.

Altre alternative con opzioni

I trader possono trovare più alternative di investimento con le opzioni. Sono altamente flessibili. Esistono molte strategie per ricreare posizioni di opzioni sintetiche.

Queste posizioni offrono agli investitori una miriade di modi per ottenere i loro obiettivi di investimento. Oltre alle posizioni sintetiche, le opzioni hanno molte altre alternative. Ad esempio, molti investitori lavorano con broker che addebitano un piccolo margine per la vendita allo scoperto di azioni. Altri trader lavorano con loro (broker) che non desiderano vendere allo scoperto.

L'incapacità di fare il rovescio della medaglia quando richiesto limita i trader e gli investitori. Tuttavia, nessun broker può pronunciarsi contro i trader che acquistano put per "giocare al ribasso". Questo è un grande vantaggio per gli investitori.

Le opzioni consentono inoltre ai trader di negoziare la "terza dimensione" del mercato. È interessante notare che possono anche scambiare in "nessuna direzione", movimenti delle azioni e durante la volatilità. Per lo più, le azioni non mostrano movimenti "grandi"; ma gli investitori hanno anche il vantaggio di operare in stagnazione. Pertanto, le opzioni possono offrire solo più alternative che possono dare loro profitto in tutti i tipi di mercati.

Perché le opzioni sono una scelta migliore?

Tuttavia, se vuoi sapere perché le opzioni sono una scelta migliore, leggi questo:

copertura

Lo scopo principale dell'invenzione delle opzioni era la copertura. Aiuta a ridurre il rischio. Quindi, prendi le opzioni come polizza assicurativa. Come se assicuri la tua auto e

home, le opzioni possono garantire il tuo investimento in caso di un movimento di caduta.

Supponiamo che un trader voglia acquistare qualcosa relativo ai titoli tecnologici. Ma vuole anche limitare la sua perdita. Il trader può fare queste semplici opzioni di rendimento, che gli danno due vantaggi: minimizzare il rischio e massimizzare il profitto. La vendita allo scoperto può anche ridurre la perdita al momento di una recessione.

Speculazione

La capacità di prevedere il prezzo futuro è speculazione, come suggerisce il nome. Si potrebbe pensare che il prezzo di un'azione scenda in un giorno, sulla base dell'analisi tecnica o fondamentale. Potrebbe vendere le azioni o vendere put dopo la speculazione.

Questo ha un'attrazione per molti investitori per le opzioni call perché offre una leva finanziaria. Un'opzione call (out of the money) può costare solo alcuni centesimi di pochi dollari rispetto al prezzo pieno del titolo di \$ 100.

Come funziona il trading di opzioni?

Quando si pesano i contratti di opzione, è importante determinare le probabilità future. Le opzioni diventano più costose quando c'è una maggiore prevedibilità in futuro. Ad esempio, quando il valore di un'azione aumenta, aumenta anche il valore della chiamata. Questo è fondamentale per capire il valore delle opzioni.

Una scadenza più breve significa un valore inferiore di un'opzione poiché le possibilità di aumento del prezzo diminuiscono man mano che la scadenza si avvicina. Se un trader acquista un'opzione di 1 mese out-of-the-money, mentre le azioni non si muovono, perde il suo valore. È perché il tempo è denaro quando si tratta di trading di opzioni. Questo spreco di opzioni è chiamato "decadimento temporale".

Allo stesso modo, se un trader acquista un'opzione con una scadenza più lunga; le possibilità di movimento del prezzo per quell'opzione diventano sempre più luminose poiché c'è abbastanza tempo perché il prezzo aumenti.

Il prezzo sale anche con la volatilità. Quando il mercato è incerto, le probabilità aumentano. Se la volatilità di un asset aumenta, le oscillazioni dei prezzi massimizzano la probabilità di movimenti sostanziali sia al rialzo che al ribasso.

Il prezzo più alto aumenta anche le possibilità che si verifichi un evento. Significa che maggiore è la volatilità, maggiore è il prezzo delle opzioni. Volatilità e trading di opzioni essenzialmente legate l'una all'altra in un certo senso.

In molte borse, un'opzione su azioni ti consente di acquistare/vendere 100 azioni. Questo

ecco perché dovresti moltiplicare il tuo premio per 100 per ottenere l'importo finale.

Dai un'occhiata a questo esempio di tabella di investimento:

	1 giugno	21 giugno	Data di scadenza
Prezzo delle azioni	\$ 67	\$ 78	\$ 62
Opzioni Prezzo	\$ 3,15	\$ 8,25	Senza valore
Valore del contratto	\$ 315	\$ 825	\$ 0
Perdita/guadagno carta	\$ 0	\$ 510	- \$ 315

La maggior parte delle volte, i titolari realizzano profitti chiudendo le loro posizioni. Significato, un titolare vende la propria opzione; mentre uno scrittore riacquista la sua posizione per la chiusura. Non più del 10% delle opzioni viene eseguito, e il 60% viene chiuso (scambiato), mentre il 30 scade senza avere valore.

Nelle opzioni, le fluttuazioni possono essere intese per "valore temporale" (valore intrinseco ed estrinseco). Il loro premio è la combinazione del valore temporale e del suo valore intrinseco. Il valore intrinseco è la somma al di sopra del prezzo di esercizio.

Il valore temporale indica il valore aggiunto che un trader deve pagare al di sopra del valore intrinseco. Questo è il valore temporale o il valore estrinseco. Pertanto, il prezzo dell'opzione nell'esempio precedente può essere considerato come:

Valore Tempo + Intrinseco = Premio
\$ 0,25 \$ 8,00 \$ 8,25

Nella vita pratica, le opzioni scambiano a (un livello superiore) il valore intrinseco. È perché le probabilità che un evento accada non possono mai essere assolutamente zero, anche se non è mai nelle carte!

Tipi di opzioni

Esistono due tipi principali di opzioni: americana ed europea. La prima tipologia può essere esercitata in qualsiasi momento compreso tra la data di acquisto e la scadenza. Inoltre, le opzioni con sede negli Stati Uniti hanno un premio più elevato. La funzionalità di utilizzo iniziale loda questo.

Ma le opzioni europee possono essere esercitate solo alla data di scadenza e in prossimità di essa. La maggior parte delle opzioni in borsa appartiene al secondo tipo.

C'è anche un altro tipo chiamato Opzioni esotiche che sono in realtà una varietà di profili di vincita dalle opzioni vaniglia. Le opzioni esotiche sono in genere pensate per

investitori professionali. Altri tipi di opzioni includono opzioni binarie, asiatiche, knockout, barriera e bermudiane.

Liquidità delle opzioni e tempo di scadenza

Esiste un altro modo per classificare le opzioni: in base alla durata. Le opzioni a breve termine scadono con 12 mesi. Le opzioni a lungo termine hanno un tempo di scadenza maggiore. Sono conosciuti come LEAPS o titoli di anticipazione di equità a lungo termine. Sono come opzioni normali con una durata di tempo generalmente più lunga.

Le opzioni possono anche essere classificate in base alla loro data di scadenza. Molti set di opzioni scadono il venerdì, ogni settimana, ogni 30 o 31 di un mese e su base giornaliera. Ci sono anche scadenze trimestrali per ETF e Index Options.

Come leggere la tabella delle opzioni?

Non è possibile fare trading di opzioni e manca il know-how per leggere le tabelle delle opzioni. Ecco come leggere la tabella delle opzioni senza difficoltà.

- Noterete un termine "Volume (VLM)" nella tabella. Indica il numero totale di contratti negoziati nell'ultima sessione.
- Sentirai anche "Offerta" e "Chiedi". Un'offerta è il prezzo più recente a quale i trader desiderano acquistare un'opzione. Mentre un "ask" è il prezzo più recente al quale il mercato desidera vendere un'opzione.
- La volatilità dell'offerta implicita (IMPL BID VOL) si riferisce all'incertezza nella velocità e la direzione del prezzo in futuro.
- Delta è la previsione o probabilità. Ad esempio, ci sono 30 percentuali di probabilità di scadenza di un delta del 30.
- Open Interest (OPTN OP) indica il totale complessivo dei contratti per a opzione specifica. OPTN OP si riduce quando il trade aperto si chiude.
- Gamma (GMM) è chiamata velocità di un'opzione. Si può anche chiamare il movimento del delta o della predicazione.
- Vega e Theta sono due valori greci utilizzati nelle tabelle di trading di opzioni. Vega rappresenta l'importo al quale è probabile che il prezzo di un'opzione cambi. Theta rappresenta il grado di variazione del valore verso il basso nel prezzo di un'opzione durante un giorno che passa.
- Il 'prezzo di esercizio' è un termine usato per il prezzo al quale qualcuno compra/vende sicurezza sottostante se desidera utilizzare le opzioni.

3. PRINCIPALI CONCETTI DI NEGOZIAZIONE DI OPZIONI

La negoziazione di azioni è un termine utilizzato in borsa la semplice definizione di negoziazione di opzioni è che "è il contratto tra due parti in cui l'acquirente (detentore) di opzioni di azioni acquista il diritto ma non l'obbligo di acquistare o vendere azioni del azioni sottostanti ad un prezzo predeterminato da/al venditore dell'opzione (scrittore) entro un periodo prestabilito.'

Le opzioni offrono sistemi alternativi che consentono all'investitore di sfruttare lo scambio e negoziare le protezioni sottostanti. Esistono diversi tipi di procedure, tra cui diverse opzioni di mix, risorse nascoste e diversi derivati.

Ti verrà in mente una domanda sul motivo per cui una persona ha bisogno del trading di opzioni, il trading di opzioni è il metodo più efficiente utilizzato in borsa e precede di un ampio margine la borsa moderna.

Quindi, non si deve pensare che sia solo una truffa creata da qualche gruppo di persone per manipolare le menti per guadagnare soldi perché ogni volta che una persona comune pensa di investire denaro in borsa rimane confusa da termini come questi, quindi cerchiamo dirti che cos'è il trading di opzioni.

Storia del trading di opzioni

Alcuni credono che siano stati i greci a dare l'idea del trading di opzioni. Molto prima del mondo moderno, gli esseri umani stavano cercando di decidere i prezzi di diversi beni, ed è così che sono stati introdotti nel mondo diversi metodi di commercio.

Rivediamo da zero qui...

Ti forniremo un semplice esempio per capire cos'è il trading di opzioni, ma per farti capire vogliamo che ti concentri sull'esempio con una mente vuota, ad esempio, vuoi acquistare azioni per \$ 4000 e vai dal broker, ma il broker ti offre un'eccitante offerta che puoi acquistare azioni per \$ 4000 ora, oppure puoi dare un token di 400 e prenotare la tua scrittura per acquistarlo a \$ 4000 dopo un mese, e anche se le azioni aumentano di valore a quella volta. Ma l'importo del token non sarà rimborsabile.

Ora pensi che sia possibile che il titolo aumenti il suo prezzo a

4020 in quel momento e puoi anche comprarlo dopo l'aumento di prezzo, e visto che hai pagato solo 400 quindi hai il resto dei soldi da usare altrove. Ora puoi aspettare facilmente un mese e decidere, riconoscendo i prezzi delle azioni dopo un mese, se vuoi acquistare il titolo o meno.

Ora, questa è una semplificazione eccessiva, e questo è il trading di opzioni. Nel mondo del trading le opzioni vengono utilizzate come strumenti, proprio come un musicista ha bisogno di uno strumento diverso per ottenere la canzone perfetta; un broker ha bisogno di opzioni per fare una vendita perfetta. E il suo prezzo è per lo più derivato dalle azioni.

Ti assicuriamo che se leggerai l'articolo fino alla fine, saprai perfettamente cos'è il trading di opzioni e ti diremo anche diverse strategie utilizzate nel trading di opzioni.

Rischi nel trading di opzioni?

La maggior parte delle strategie utilizzate dagli investitori in opzioni ha un rischio limitato ma anche un profitto limitato, quindi il trading di opzioni non è un metodo che ti renderà ricco durante la notte.

Il trading di opzioni potrebbe non essere adatto a tutti i tipi di investitori, ma sono tra le scelte di investimento più flessibili.

Le opzioni di investimento sono molto probabilmente utilizzate per ridurre il rischio di un calo dei prezzi delle azioni, ma ogni tipo di investimento comporta un piccolo rischio. I rendimenti non sono mai garantiti, gli investitori cercano opzioni per gestire i rischi per limitare una potenziale perdita.

Gli investitori possono scegliere di prendere delle opzioni perché la perdita è limitata al prezzo da pagare per il denaro simbolico. E in cambio, ottengono il diritto di acquistare o vendere azioni lì al loro prezzo desiderabile, quindi le opzioni nel trading avvantaggiano molto gli investitori.

Diverse strategie utilizzate nel trading di opzioni

I trader spesso sanno molto poco delle strategie utilizzate nel trading di opzioni e passano alle opzioni di trading, sapendo che le diverse strategie possono ridurre il rischio di potenziali perdite sul mercato e gli operatori possono anche imparare a limitare il rischio e massimizzare il rendimento. Quindi, con un piccolo sforzo, i trader possono imparare a sfruttare appieno la flessibilità e il potere che le opzioni su azioni possono fornire.

Stock VS Opzione

Bisogna pensare che perché c'è bisogno di scambiare opzioni quando qualcuno

può anche commerciare semplicemente questo pensiero confonde molti di noi, quindi ecco la risposta

Il contratto di opzioni ha una data di scadenza, a seconda del tipo di opzioni che stai utilizzando. Può essere in settimane, mesi o addirittura anni a differenza delle scorte, perché le scorte non hanno data di scadenza.

Le azioni sono generalmente definite da numeri, ma d'altra parte non ci sono numeri nelle opzioni.

Le opzioni guidano il loro valore da qualcos'altro. Ecco perché rientrano nella categoria dei derivati, a differenza delle azioni.

I proprietari di azioni hanno il loro diritto nella società (dividendo o voto) d'altra parte le opzioni non hanno diritto nella società

Alcune persone potrebbero avere difficoltà a comprendere il metodo delle opzioni sebbene l'abbiano seguito anche nell'altra transazione, ad esempio (assicurazione auto o mutui).

Piattaforme di trading di opzioni

se una persona vuole negoziare opzioni, deve avere un conto di intermediazione e, per questo, vorrà capire cosa fa prima di registrarsi con un broker. Ogni piattaforma è unica e ha i suoi pro e contro. Quindi una persona deve saperne di più sulla migliore piattaforma di trading di opzioni per determinare quale potrebbe essere la più adatta alle proprie esigenze.

Se una persona desidera trovare la migliore piattaforma di trading, deve esaminare diversi broker e piattaforme di trading di opzioni. Una persona deve considerare diversi fattori come i prezzi competitivi, l'esperienza high tech, adatta a una varietà di esigenze e stili dei trader.

Alcune delle migliori piattaforme di trading di opzioni per il 2020

sono: TD Ameritrade: Best Overall

Tastyworks: secondo classificato

Charles Schwab: il migliore per i principianti Webull: il

migliore per senza commissioni Interactive Brokers: il

migliore per i trader esperti

Metodo di pratica delle opzioni

1. Le azioni vengono acquistate e l'investitore vende opzioni call sulle stesse azioni che ha acquistato. Il numero di azioni che hai

acquistato dovrebbe corrispondere al numero di opzioni call che hai venduto.

2. Dopo aver acquistato le azioni, l'investitore acquista opzioni put per ottenere quote uguali. Married agisce come una polizza assicurativa contro le perdite a breve termine opzioni call con un prezzo di esercizio specifico. Allo stesso tempo, venderai opzioni call simili a un prezzo di esercizio più elevato.

3. Un investitore acquista un'opzione con contanti dall'esterno, mentre contemporaneamente opera un'opzione call out of the cash per un'azione simile.

4. L'investitore acquista contemporaneamente un'opzione call e una scelta put. Le due alternative dovrebbero avere un costo di esercizio e una data di scadenza simili.

5. L'investitore acquista l'opzione call in contanti e la scelta put contemporaneamente. Hanno una data di scadenza simile; tuttavia, il loro costo di sciopero è straordinario. La spesa dello sciopero dell'informazione non dovrebbe essere esattamente la spesa dello sciopero della chiamata

Strategie per il trading di opzioni

I trader di opzioni utilizzano diverse strategie per trarre profitto da questo business. I diversi modi di strategie vengono utilizzati per ottenere profitto, il che implica l'utilizzo di molte alternative e combinazioni. Le strategie più comuni sono chiamate coperte, condor di ferro, acquisto di chiamate e acquisto di put. Il trading di opzioni fornisce strategie avanzate.

Acquisto di chiamate

L'acquisto di call o la strategia long call viene utilizzata quando un investitore acquista sempre più call e imposta un'opzione sull'esatto asset sottostante con data e prezzo fissi. L'investitore utilizza questa strategia quando si sente rialzista e fiducioso in un aumento del prezzo delle azioni. In questo tipo, l'investitore aumenta il rischio in quanto può affrontare un enorme profitto o perdita, ma non si sa sempre in che direzione andrà il titolo.

L'acquisto mette

Di solito, l'investitore utilizza questa strategia quando è ribassista su alcune azioni; ad esempio, l'investitore è fiducioso in un determinato titolo e ha una buona conoscenza del titolo, ma non vuole correre un rischio enorme, quindi utilizza la strategia di vendita allo scoperto.

L'opzione put ottiene un aumento di valore quando il prezzo dell'attività diminuisce. Quando il mercato scende, il profitto aumenta nelle vendite allo scoperto. Il rischio non è confermato in quanto i trade tornano con la leva finanziaria. Ma nella remota possibilità che il

base Ascent oltre i prezzi dell'opzione, l'opzione scadrà inutilmente.

Chiamate coperte

Questa strategia prevede una piccola variazione del prezzo, il profitto non è così grande, ma il rischio che comporta è inferiore. La call coperta acquista 100 azioni e poi vende un'opzione call ogni 100 azioni. La strategia di chiamata coperta offre una possibilità di profitto all'investitore e riduce anche il rischio. L'azione è protetta da questa chiamata quando il prezzo dell'azione diminuisce.

Condor di ferro

In questa strategia, il trader vende una put, ne compra un'altra a un prezzo basso e le usa per acquistare una call e poi venderla a un prezzo alto dopo un po' di tempo. Se il prezzo delle azioni viene mantenuto da qualche parte tra due put o call, allora otteniamo un profitto. La perdita arriva con possibilità, una se il prezzo aumenta improvvisamente e l'altra è se il prezzo diminuisce improvvisamente, è lo spread che causa questa condizione. Questa strategia è utilizzata da trader neutrali o in un luogo neutrale.

Vengono utilizzate diverse altre strategie, che sono:

- Farfalla spezzata
- Farfalla di ferro
- Lucertola di giada
- Chiamata dell'orso
- Calendario diffuso
- Rivestimento protettivo

Il valore del trading di opzioni

Quando acquistiamo una bici o un'auto, vogliamo proteggerli; l'assicurazione serve per la sicurezza dell'auto. Quindi, proprio come l'assicurazione, l'opzione ci dà sicurezza. Investiamo denaro e acquistiamo azioni ora vogliamo proteggere il nostro investimento, per questo usiamo le opzioni.

L'opzione ci fornisce una buona protezione dei nostri soldi. Per esempio:

Abbiamo comprato le 100 azioni al prezzo di 150 dollari, che valgono 15000. Abbiamo investito e ora corriamo il rischio di una diminuzione del prezzo. Compriamo l'opzione per rimuovere il rischio dalle nostre spalle e il ragazzo viene pagato ora si assume il rischio. Compriamo l'opzione put per 500 dollari. Se lo stock aumenta con il tasso di 170, otterremo il profitto, anche acquistando un'opzione di 500.

Ma se il prezzo delle azioni diminuisce con il tasso di 130, siamo ancora stabili, il

la perdita non ci riguarderà poiché abbiamo il contratto sull'opzione che ha lo stesso prezzo. Possiamo vendere le azioni allo stesso tasso che abbiamo acquistato, quindi in tal caso, utilizzando il commercio di opzioni, la possibilità di perdita è molto bassa. Fornisce molti modi per ottenere profitti nel trading.

Trading di opzioni VS Trading di azioni

Un trading di opzioni non è trading di azioni. Bene, entrambi fanno trading, ma sono molto diversi. Molte persone non conoscono nemmeno il trading di opzioni; è solo un altro tipo di trading. Poche cose fanno la differenza tra il trading di opzioni e il trading di azioni, ecco questi punti.

- Nel trading di opzioni, il valore viene preso da qualcun altro e ha un contratto con esso. Non ottiene i valori da solo. Questo è completamente diverso dal trading azionario. Il trading di opzioni appartiene alla categoria dei derivati.
- In stock, i numeri sono definitivi, ma un'opzione, i numeri non sono definitivi.
- Il trading di opzioni utilizza il contratto, che ha la data di scadenza, la persona non ha significato dopo la data di scadenza. La data può essere in mesi o anni a seconda dell'opzione utilizzata. La compravendita di azioni non ha date di scadenza.
- Nel trading di opzioni, il proprietario non ha alcun diritto sulla società. Non hanno relazioni di alcun tipo relative alla società. In azioni, avevano i diritti sulla società.

Rischio nel trading di opzioni

Il rischio che comporta l'opzione non è tanto quanto la gente pensa. Il trading comporta dei rischi. Le sue procedure lavorano insieme al rischio. In opzione, il rischio può contenere solo in poche cose.

- Il trading di opzioni utilizza molte strategie con queste strategie. Ognuno ha il suo rischio. Le poche opzioni funzionano sul processo di aumento e diminuzione spontaneo. Questo a volte dà una grande perdita per l'investitore.
- L'opzione comporta molta complessità. Questo trading è difficile da capire. Le strategie di per sé sono complesse. Coloro che sono alle prime armi nell'opzione non la capiscono bene e investono i soldi con la poca conoscenza che si traduce in una perdita.
- L'altro problema con il trading di opzioni è che ha una scadenza

data, che può farti investire tutto se il contratto scade. Questo è un grande fattore in questo commercio.

Conto opzioni

Per il commercio di opzioni, hai bisogno di un conto di opzioni. Prima di creare un account, è necessario compilare l'accordo con il proprio broker. Il broker conoscerà il tuo investimento e il tuo commercio. Genererà la strategia in base al livello di trading che desideri. Il broker ti guiderà sul trading di opzioni e sulle sue politiche.

Che cos'è un conto opzioni?

Il conto che viene utilizzato per accedere al commercio di opzioni è un conto di opzioni. Il broker dà accesso all'utente per un conto opzioni. Per tutto il trading viene utilizzato il conto di intermediazione; fa la vendita e l'acquisto. Dopo aver fornito tutti i dettagli, sarai in grado di aprire l'account.

Il broker indica i due tipi di conto che si desidera aprire, il conto in denaro reale e il conto in denaro demo. Dopo tutte le procedure, il tuo account sarà pronto per il trading.

Il broker giusto

Prima di scegliere il tuo broker, fai il check-in su di lui. Scegli sempre quello con una fonte autentica. Le informazioni che gli fornisci dovrebbero essere protette. Controlla sempre il pagamento e il suo costo. Punta all'opzione giusta.

Il miglior broker

- Schwab Brokerage (\$ 0,65 per contratto di opzioni)
- Commercio elettronico (\$ 0,65 per contratto di opzioni)
- Ally Invest (\$0,5 per contratto scambiato)
- TD Ameritrade (\$ 0,65 commissione per contratto)

Le migliori piattaforme

Il trading di opzioni è un rischio di alto livello. Ha bisogno di essere protetto dalle frodi. Quando si seleziona la piattaforma, è necessario selezionare la migliore. Ci sono molte migliori piattaforme disponibili sul mercato; c'è una buona reputazione, come.

- Charles Schwab, questa piattaforma è la migliore per i principianti. Se sei un principiante, dovresti scegliere questa piattaforma. Questa piattaforma offre di più

comprensione agli utenti.

- TD Ameritrade è la migliore piattaforma al mondo per il trading di opzioni. Il costo è basso e nessun requisito minimo di account
- Tastyworks fornisce l'accesso commerciale a diversi dispositivi. Consente PC, laptop e telefoni cellulari. È una delle piattaforme più high-tech.

La piattaforma Webull non dà commissioni

Il trading di opzioni non è pensato per i principianti che non hanno idee sul mercato. Quindi, se stai appena iniziando il tuo viaggio con il mercato azionario, potresti dover dedicare del tempo all'apprendimento dei concetti di base del trading di opzioni.

Quando si parla di azioni, si tratta di investimenti e di trasformare quell'investimento in profitto. Quindi richiede una forte conoscenza ed esperienza per realizzare grandi profitti ed evitare perdite.

4. MITI E CONCETTI DIFETTOSI SUL TRADING DI OPZIONI

T qui ci sono molti miti e malintesi legati al termine "Opzioni Trading" non solo nel mercato azionario ma anche per il pubblico in generale. Il trading di opzioni è noto per essere rischioso; secondo Mike Bellafiore, il co-fondatore del trading CMB, "Il trading è uno sport di sopravvivenza, reinvenzione e perseveranza, anche per il trader di successo".

In effetti, nel mercato azionario o negli affari, non esiste una cosa come l'assicurazione; c'è sempre un rischio nel mettere i tuoi soldi in qualcosa. Il mercato azionario o il business sono sempre una questione di numeri e buone strategie. Se le tue strategie e i tuoi numeri sono giusti, ci sei dentro per il lungo periodo; altrimenti non ti ritroverai con niente.

Il vincitore dell'US Investing Championship nel 1984 Martin S. Schwartz afferma:

"Molte persone sono così invischiare nei mercati da perdere la prospettiva. Lavorare più a lungo non significa necessariamente lavorare in modo più intelligente. Anzi, a volte è il contrario". In un'attività commerciale, puoi utilizzare i tuoi difetti o fallimenti anche per i tuoi benefici futuri; secondo Brett Steenbarger, un trader attivo e un Ph.D. studioso, "non saremo mai perfetti come trader. Questo è ciò che ci fa continuare a imparare, in continua crescita. La nostra sfida principale è quella di utilizzare le nostre carenze come ispirazione, alimentando il miglioramento continuo".

Ci sono molti miti e idee sbagliate sul trading.

Idea sbagliata n. 1 Il trading e il gioco d'azzardo sono la stessa cosa

Il primo e il più comune è il trading e il gioco d'azzardo sono le due facce della stessa medaglia. Nel trading, un trader passa attraverso tutti i dati presenti, passati e numeri se il gioco d'azzardo è un gioco di probabilità disponibili.

Il trading riguarda l'analisi tecnica in cui si esaminano i dettagli, i rischi, il profitto, il guadagno e il mercato, mentre il gioco d'azzardo si basa su valori fondamentali. Metti i tuoi soldi in quello che potresti pensare che accadrà. Inoltre, il gioco d'azzardo è una dipendenza, illegale e anche molto tossico per la salute mentale e il comportamento.

Idea sbagliata n. 2 Dovrebbe investire solo in opzioni Call e Put

Le opzioni sono il tipo di contratto che consente all'acquirente di acquistare o vendere l'attività sottostante. Per semplificare, il trader acquista un'opzione call se si aspetta che la domanda dell'attività sottostante aumenti entro un certo termine, e il trader opta per un'opzione put se si aspetta che la domanda dell'attività sottostante rientri in un determinato periodo di tempo.

È un equivoco sull'opzione call e put che sia l'unico modo redditizio di negoziare opzioni, ma in realtà l'acquisto di call e put è altamente rischioso nel trading perché non si può mai essere sicuri della domanda dell'attività sottostante per che è necessaria l'analisi corretta della direzione in cui si sta muovendo, del suo arco di tempo e della dimensione del movimento. Puoi analizzare correttamente le dimensioni e la direzione, ma se le tue opzioni sono scadute prima che avvenga il movimento, potresti perdere denaro.

Idea sbagliata n. 3 La vendita di opzioni è più redditizia

La vendita di opzioni consiste sostanzialmente nel dare a qualcuno il diritto, ma non l'obbligo, di farti acquistare 100 azioni di un titolo a un prezzo di esercizio prima della data di scadenza. In termini più semplici, in pratica pagano per aumentare la loro flessibilità e tu paghi per diminuire la tua flessibilità. Quindi, quando vendi opzioni, non stai solo usando i soldi nella tua intermediazione, ma sei anche indebitato. La vendita di opzioni può essere redditizia se si rispettano le regole, ma comporta anche un alto rischio.

Idea sbagliata #4 L'opzione Put scade senza valore

L'opzione che scade senza valore è quando le opzioni scadono dal tuo conto di trading e cessano di esistere. Ci sono molte idee sbagliate circa il 90% delle opzioni in scadenza, ma secondo il rapporto di The Chicago Board Options Exchange (CBOE), circa, solo il 30% delle opzioni scade senza valore. Il 60% delle posizioni in opzione viene chiuso prima della scadenza e il 10% delle opzioni viene esercitato.

In secondo luogo, le opzioni che scadono senza valore funzionano solo contro gli acquirenti di opzioni, ma gli autori delle opzioni ottengono comunque il loro profitto se l'opzione put è scaduta.

Idea sbagliata n. 5 Il trading di opzioni è a somma zero

Questo è uno dei miti più antichi sul trading di opzioni. Dice che se un acquirente vince, il venditore deve perdere. Ma no, non è affatto vero!

Vengono fornite opzioni per gestire il rischio. Non ti danno nulla di valore se non la possibilità di acquistare o vendere beni. Quando usi le opzioni per

copri il tuo rischio, stai trasferendo il tuo rischio a qualcun altro che è disposto a trattenerlo. Quindi il trading di opzioni non è un gioco a somma zero.

Idea sbagliata #6 Il trading di opzioni è facile

C'è un altro equivoco sul trading di opzioni che la gente presume sia facile. Secondo il commerciante e autore Charles Faulkner, afferma: "Dopo anni di studio sui trader, il miglior predittore di successo è semplicemente se la persona sta migliorando con il tempo e l'esperienza" sono necessari anni di esperienza, per conoscere completamente il mercato e le sue risorse e strategie allora puoi essere un trader di successo.

Il trading è raramente basato sulla "fortuna", è tutto basato su una buona conoscenza pratica. La maggior parte delle persone non ha una conoscenza approfondita del trading di opzioni o del mercato azionario. È più che investire denaro. Tutti gli esperti nel commercio hanno anni di esperienza e conoscenza e usano persino i loro fallimenti come arma per il loro successo futuro.

Idea sbagliata n. 7 Trading in conto fiscalmente differito

C'è un malinteso comune sull'utilizzo di conti tradizionali o IRA per l'opzione di trading perché entrambi alleggeriranno il vantaggio fiscale e potrebbe essere un piano pensionistico perfetto, ma ci sono alcuni limiti. Puoi investire solo fino a un certo limite tramite il tuo account con imposte differite. Non puoi usare i soldi prima della pensione. Dopo cinque anni, puoi solo prelevare il reddito.

5. STRATEGIE DI TRADING CON LE TOP OPTION

T Il trading di opzioni utilizza alcune strategie per gli specialisti finanziari per trarre vantaggio dal trading. I vari metodi di strategie sono utilizzati per ottenere profitto, che include l'utilizzo di numerose altre opzioni e combinazioni.

Lo scambio alternativo offre tecniche avanzate. Questi sistemi aiutano gli specialisti finanziari ad aumentare i vantaggi più estremi. La strategia superiore viene utilizzata per diversi livelli di trading. Molte strategie di trading popolari sono utilizzate nel mercato. Queste strategie sono ben note nel trading e hanno un numero di utenti. Le seguenti sono le migliori strategie di trading di opzioni.

Acquisto di chiamate

La metodologia di acquisto delle chiamate o delle chiamate lunghe viene utilizzata quando uno specialista finanziario acquista progressivamente le chiamate e imposta una scelta sulla specifica risorsa nascosta con data e costo fissi. L'investitore si aspetta una leva maggiore rispetto al semplice possesso del titolo. Lo specialista finanziario utilizza questa procedura quando si sente rialzista e sicuro di un incremento a un certo costo delle azioni. La fiducia dell'investitore li prepara a scegliere questa strategia di opzione.

In questo tipo, gli speculatori aumentano il rischio in quanto possono affrontare il gigantesco beneficio o perdita, ma è costantemente oscura la direzione in cui va il titolo. Se il titolo sale, l'investitore otterrà il profitto e se il prezzo delle azioni diminuisce, l'investitore dovrà affrontare una grande perdita. Anche i trader più esperti ad un certo punto subiscono una perdita, ma ciò non significa che daranno solo perdite.

Il profitto in esso di solito copre la perdita degli investitori. Ad esempio, se l'investitore desidera acquistare una casa in via di sviluppo, acquisterà l'opzione di acquisto call, in questo modo può pagare lo stesso prezzo in base al contratto poiché il costo di una casa è di 200.000 ora come quando lo sviluppo sarà completo, il costo della casa sarà aumentato.

Supponiamo che il prezzo della casa sia aumentato incredibilmente dopo 2,3 anni e che ora la casa valga 400.000. L'investitore pagherà lo stesso importo, anche se il suo prezzo sul mercato è doppio a causa dell'opzione che può pagare il

stesso importo di 200.000 come scritto sul contratto. L'unica cosa di cui l'investitore si preoccuperà è la data di scadenza.

L'acquisto mette

In questa strategia di trading di opzioni, l'investitore ha il diritto legale di vendere le azioni al prezzo indicato. Anche la data è fissata ad una certa ora. Le put di acquisto danno più autorità all'investitore.

Normalmente, lo specialista finanziario utilizza questa strategia quando è ribassista su alcune azioni; per il modello, lo speculatore è positivo riguardo al titolo specifico e ha una grande comprensione del titolo, ma non vorrebbe affrontare una sfida immensa, quindi utilizza la procedura di vendita allo scoperto.

La scelta put ottiene un aumento di stima quando il costo del beneficio diminuisce. Quando il mercato scende, il vantaggio aumenta con la vendita allo scoperto. La possibilità non è affermata come il ritorno commerciale con leva. Ma se l'Ascesa fondamentale supera il valore di scelta, allora l'alternativa terminerà inutilmente.

Per lo più i trader sono sicuri che il mercato cadrà. Acquistano l'azione per venderla in un determinato momento in quanto ne hanno il diritto, i valori dell'azione aumentano quando il titolo si muove nella direzione opposta. È un modo semplice per guadagnare.

Se vuoi l'assicurazione sulle tue azioni in stock, puoi acquistare un'opzione put. Se l'investitore avesse una quota di 500 dollari, e si rendesse conto che il mercato avrebbe perso il valore. Possono vendere la loro quota a un prezzo ridotto di circa 475 dollari. La perdita si riduce in questo modo.

Chiamata coperta

Questa strategia prevede una piccola variazione del prezzo, il profitto non è così grande, ma il rischio che comporta è inferiore. La call coperta acquista 100 azioni e poi vende un'opzione call ogni 100 azioni. La strategia di chiamata coperta offre una possibilità di profitto all'investitore e riduce anche il rischio.

L'azione è protetta da questa chiamata quando il prezzo dell'azione diminuisce. In questa strategia, il venditore dell'opzione call possiede il relativo importo dello strumento sottostante. Nell'opzione call coperta, puoi acquistare azioni e vendere l'opzione call su Out of Money (OTM).

Il termine buy-write viene indicato quando la call è stata venduta contestualmente all'acquisto di azioni. Riceviamo una piccola somma per le vendite di chiamata quando paghiamo

per azioni. Dobbiamo vendere le chiamate per generare reddito in esso. Questa opzione richiede investimenti diretti e richieste di vendita.

L'investitore utilizza questa strategia di opzione quando pensa che il prezzo non aumenterà ulteriormente. Dobbiamo mantenere una posizione lunga. La possibilità di profitto è bassa poiché l'aumento del prezzo non è previsto.

Supponiamo che il titolo fosse scambiato a \$ 200 il 20 maggio 2014; The Leg 1 Acquista 100 porzioni di azioni per \$ 200 e Leg 2: Vendi i 206 strike 26 giugno 2014 Chiama per \$ 7,30, dimensione del lotto - 100 offerte

La somma pagata per assumere queste due posizioni si avvicina - il costo delle azioni pagato meno il premio di chiamata ha ottenuto, ad esempio, \$ 20.000 (acquisto di azioni) - \$ 730 (Premium ottenuto) = \$ 19.270

Se il valore delle azioni supera il call strike di 206, verrà calcolato e le azioni verranno vendute. In ogni caso, la metodologia porterà un vantaggio poiché sei protetto dalle azioni che possiedi.

Stato, il costo delle azioni alla fine è di \$ 210.

Nel caso in cui il titolo scenda al di sotto del prezzo del titolo sottostante, la tua posizione lunga è in una sfortuna. Tuttavia, hai qualche pad dal premio di chiamata ottenuto vendendo la chiamata.

Stato, il titolo scende ed è a \$ 190 alla scadenza

Condor di ferro

In questa strategia, il trader vende una put, ne compra un'altra a un prezzo basso e le usa per acquistare una call e poi venderla a un prezzo alto dopo un po' di tempo. Se il prezzo delle azioni viene mantenuto da qualche parte tra due put o call, allora otteniamo un profitto.

La perdita arriva con possibilità, una se il prezzo aumenta improvvisamente e l'altra è se il prezzo diminuisce improvvisamente, è lo spread che causa questa condizione. Questa strategia è utilizzata da trader neutrali o in un luogo neutrale. Questa è una semplice strategia di opzioni.

L'opzione Iron Condor non richiede un grande investimento per avviare un commercio; puoi iniziare lo scambio con un importo minimo. L'investitore fa affidamento sul titolo per rimanere in un punto particolare. È una piccola strategia che comporta rischi, ma l'investitore investe in una piccola quantità per mantenere il rischio.

Considera che il titolo viene scambiato al costo di \$ 120, eseguendo una procedura di trading Iron Condor: Vendi \$ 100 Strike Put per \$ 3.0 Vendi \$ 140 Strike Call

per \$3.0

Con l'aspettativa che il costo rimanga all'interno di questi due costi di sciopero che abbiamo prenotato, quindi otteniamo un vantaggio. In ogni caso, a causa del pericolo di sconfinare disgrazie, assicureremmo le nostre situazioni con Buy Strike Put per \$90 Buy Strike \$160 Call per \$ 2.

orso chiama

Questa opzione funziona sulla procedura di vendita e acquisto. L'investitore vende l'opzione call e quindi acquista le chiamate a un tasso di esercizio elevato. Questa opzione utilizza l'investimento del trader per ottenere il reddito di profitto. La procedura funziona su livelli limitati. Usa due gambe. Questi lavorano su un rapporto 1:1 per rendere il credito netto.

Anche se questa non è una strategia ribassista, si realizza quando si è rialzisti. Generalmente è impostato per un "credito netto" e la spesa per l'acquisto di scelte call è finanziata dalla vendita di una scelta call "in contanti". Per questa strategia di trading di opzioni, è necessario garantire che le alternative Call abbiano un posto con una scadenza simile, l'equivalente risorsa nascosta e la proporzione sia mantenuta.

Esso, per la maggior parte, assicura l'inconveniente di un Call venduto salvaguardandolo, ad esempio, acquistando un Call di costo strike più elevato. È fondamentale, tuttavia, eseguire il sistema proprio quando si è convinti che il mercato si muoverà sostanzialmente al rialzo.

L'investitore si aspetta una piccola diminuzione del titolo. Vendono le chiamate e poi acquistano le chiamate ad alto sciopero. L'opzione funziona meglio quando la volatilità è alta. La data di scadenza è abbastanza buona per gestire le cose. I trader non sprecano la data di scadenza, poiché è importante per il trading. L'investitore utilizza il lungo termine per eseguire il processo.

Se il mercato si aspetta l'aumento delle azioni, i trader vendono una chiamata di sciopero e poi ne acquistano un'altra allo sciopero più alto. L'investitore ottiene il profitto per l'importo del costo.

Se le azioni si aspettano un rialzo improvviso, allora vendono la chiamata e poi acquistano quelle nuove che acquistano anche a un tasso elevato, ma invece della perdita ottengono un profitto e quindi acquistano più chiamate.

lucertola di giada

In questa strategia di trading di opzioni, i trader vendono short call e put, e il

le attività sottostanti non dovrebbero muoversi. Il costo incassato nei risultati è ottimo. Tutte le opzioni hanno la stessa data di scadenza. Riduce al minimo il rischio e massimizza la ricompensa. Le opzioni di trading massimizzano il rischio in una direzione.

L'opzione lucertole di giada è una sorta di strategia di trading di opzioni che viene provata dai trader per trarre vantaggio dai loro scambi

Se dovesse verificarsi un evento di Straddle e Strangles, i Lizard riducono il rischio di rialzo. Sono più preziosi quando la base rimane o fluttua verso lo sciopero. Elevati benefici vengono creati in situazioni IV alto e non ribassiste. Questa strategia neutrale prevede call short e spread short put. È una strategia lenta; non aumenta improvvisamente. Utilizza il costo della chiamata e pone il costo ad alta volatilità.

Supponiamo che l'investitore accetti che questo commercio sia una possibilità di svantaggio. Nel caso in cui il titolo ABC si sposti sopra £ 25 per azione, lo specialista finanziario perderebbe \$ £ 1 sullo spread call, ma guadagna £ 1,10 dal premio raccolto per un'aggiunta netta di £ 0,10.

L'investitore beneficia dello scambio, a meno che il costo di ABC si sposti al di sotto del costo di esercizio del bare put di più del massimo che viene raccolto. In questo modello, il costo delle azioni dovrebbe scendere sotto £ 18,90.

Opzione collare

È simile alla chiamata coperta ma ha put protettivi extra per proteggere il valore della sicurezza tra 2 limiti. La Strategia di Trading di Opzioni Collar può essere costruita tenendo porzioni di quelle nascoste per tutto il tempo e acquistando alternative put-call e vendendo scelte call contro le offerte mantenute.

Si può sostenere l'inconveniente atteso nelle offerte acquistando il fondamentale e, allo stesso tempo, acquistando un'alternativa put al di sotto del costo corrente e vendendo una scelta call al costo corrente. Acquista un'opzione put piuttosto che abbassare il limite per la protezione; vendere l'opzione call al limite superiore.

Entrambi devono avere la stessa data di scadenza e la stessa quantità. Le opzioni call e put sono esaurite. Il prezzo delle attività sottostanti è scaduto al prezzo di esercizio dell'opzione call short. L'instabilità è sorprendente quando il mercato è instabile al punto in cui il costo di un'ascesa alternativa è probabile che il costo possa diminuire e si possa perdere il beneficio.

In tal caso, il vantaggio dovrebbe essere assicurato. L'opzione protegge molto le perdite e riduce le possibilità di tutte le perdite, ma con tutta la protezione,

a volte riduce il profitto.

Supponiamo che il valore delle azioni salga a Rs 50 In questo caso; il trader avrebbe compreso che la stima della sua detenzione di azioni sarebbe salita a $(100 \times 50) = \text{Rs } 5000$.

Essendo il venditore dell'opzione Call, ha anticipato che il costo dei fondamentali dovrebbe scendere. In ogni caso, il suo costo è sicuramente aumentato. L'acquirente dell'opzione Call eserciterà il suo privilegio e acquisterà l'alternativa Call al costo di esercizio di 48, che è inferiore al costo del fondamentale che è 50. Quindi il venditore dell'opzione ha ottenuto $(48 \times 100) = \text{Rs } 4800$ vendendo il Opzioni di chiamata.

Per un acquirente alternativo Put, un'opzione è in contanti se il costo di esercizio è superiore al costo dell'occulto. Per questa situazione, poiché il costo d'esercizio di 43 non è esattamente il CMP dell'occulto, che è 50, e in questo senso l'opzione è resa inutile per lui.

Beneficio netto dello scambio = $\text{Rs } 5000 - \text{Rs } 4800 + 500 - 300 = 400$

Diffusione diagonale

La strategia dell'opzione con spread diagonale utilizza molti strike e mesi. Funziona con i bit combinati di un call spread lungo e uno short call spread. La diffusione diagonale si sposta in diagonale e anche i nomi.

L'opzione è presentata in diverse righe e colonne. In questa strategia di trading di opzioni, vengono venduti i termini brevi e vengono acquistati i termini lunghi. Una mancanza transitoria o Forza che pensi possa salire o scendere ancora una volta, a quel punto, a suo vantaggio.

Il sistema è controllato sul lato corto per il rischio e, se il mercato funziona senza intoppi, può diventare aperto sul lato lungo.

Nel momento in cui viene eseguito per contanti, consente di soddisfare le esigenze del bordo. L'investimento è ad alto rischio quando funziona rapidamente a modo nostro. Lo spread diagonale ha il suo setup, che dobbiamo seguire.

- È richiesto lo stesso numero di opzioni.
- Le opzioni devono avere l'esatto titolo sottostante.
- Le opzioni nello spread diagonale dovrebbero avere la stessa classe.
- Vengono utilizzate le diverse date di scadenza.
- I due diversi prezzi di esercizio.

Nello spread diagonale, lo spread diagonale call long rialzista acquista l'opzione con il tasso di esercizio più basso e la data di scadenza più lunga, quindi vende l'opzione con data corta con tassi di esercizio elevati.

La farfalla

L'opzione farfalla ad ala spezzata è un po' simile alla strategia di trading a farfalla, ma in questa strategia di trading, le chiamate e le put sono molto simili alla strategia direzionale piuttosto che alla strategia a farfalla.

I suoi lati hanno un diverso livello di rischio; il rischio è diverso da una parte all'altra. Di solito, il profitto si verifica se il sottostante scade al prezzo di esercizio short. L'opzione della farfalla ad ala spezzata fornisce maggiori profitti rispetto all'opzione della farfalla.

Futuresmag merita il merito di aver generato la "Broken Wing Butterfly", un'opzione straordinaria in contrasto con la Butterfly, dove l'obiettivo è avviare uno scambio a costo zero.

È un fantastico trading di opzioni che espande gli attributi positivi di un Butterfly Spread. A differenza della Long Butterfly, in cui è necessario pagare un altro addebito, la strategia Broken Wing Butterfly è una procedura di credito netto, spesso messa a punto per costruire la probabilità di beneficio.

La strategia Broken Wing Butterfly è equivalente a una Butterfly in cui lo spread venduto è regolarmente più ampio dello spread acquistato. Ha la somiglianza di una lunga farfalla che ha gli stessi colpi che non sono molto diversi dallo short strike. Funziona quando l'opzione ha tutte le put o tutte le chiamate.

Ad esempio, il titolo viene scambiato a Rs100. Compri una chiamata da 120 su ABC, vendi due chiamate da 105 su ABC e acquisti una chiamata da 100 su ABC, quindi ottieni il credito netto di Rs. 10.

6. MASSIME QUALITÀ DI UN COMMERCIANTE DI OPZIONI DI SUCCESSO



'La chiave per il successo nel trading è la disciplina emotiva. Se l'intelligenza fosse la chiave, ci sarebbero molte più persone che fanno soldi con il trading' - Victor Sperandeo

T Per essere un trader di opzioni, sono richieste alcune qualità che non sono affatto difficili da raggiungere. Per sviluppare queste qualità, devi conoscere il trading di opzioni.

Trading di opzioni

Nel trading di opzioni, l'acquirente ha il diritto, quando vuole comprare (il caso di una call) e quando vuole vendere (il caso di una put), ma non è obbligato ad acquistare o vendere il determinato asset a un determinato prezzo, come suggerisce il nome 'Opzioni'. Inoltre, il trader non è obbligato a negoziare in un momento specifico. Ha la totale scelta di cosa e quando vuole fare trading.

Perché le persone tendono a fare trading di opzioni?

Le opzioni possono fornire un reddito migliore rispetto a qualsiasi altro lavoro. Non è molto diverso dalla borsa. È proprio come la tua attività; tutto ciò che devi fare è prevedere dove stanno andando il mercato e le quotazioni azionarie. Devi correre un rischio, ma il reddito sarà più alto di quanto pensassi. I risultati finali possono dare uno shock poiché i tassi di mercato continuano a cambiare ogni minuto.

Le migliori qualità di un trader di opzioni di successo

Proprio come in qualsiasi altra attività, c'è un enorme rischio di perdita. Ogni altra persona non può essere un buon uomo d'affari. Molte persone hanno un livello di pazienza debole, perdono il cuore e, in caso di fallimento, lasciano l'attività o vendono a un prezzo basso senza fare un altro tentativo.

Chi lo compra a un prezzo basso lo porta ad un altro livello. Allo stesso modo, non tutti possono essere buoni trader. Essere un trader di successo richiede determinate qualità. Se raggiungi queste qualità, nulla può impedirti di diventare uno dei trader di opzioni di maggior successo.

1. Controllo sulle emozioni
2. Conservazione dei registri
3. Trovare la strategia giusta per te
4. Coerenza
5. Imparare dai fallimenti
6. Interpretazione delle notizie
7. Essere te stesso
8. Pazienza
9. Flessibilità
10. Gestione del rischio

Queste sono alcune qualità per renderti un trader di successo nel trading di opzioni e un atteggiamento positivo nei confronti di queste abilità può renderti un trader di opzioni professionale. Scaviamo un po' di più in queste qualità per perfezionare un po' di più la tua personalità.

Controllo sulle emozioni

Mescolare le tue emozioni con i tuoi affari può portarti verso la distruzione. Devi gestire la tua vita sociale insieme a quella reale senza aggrovigliarli l'uno con l'altro. Devi essere in grado di gestire l'accaduto nella vita reale e gli avvenimenti nel business. Devi avere il controllo totale sulla tua mente e tenere i nervi saldi mentre fai gli affari.

Gestione dei record

Se tieni un registro di tutto ciò che fai, la prossima volta sarai in grado di evitare l'errore che hai commesso in precedenza e sarai in grado di vedere dove hai sbagliato.

Questa abitudine ti fornirà informazioni sulla tua ricchezza per migliorare le tue probabilità di successo. Mantenendo i registri, sarai in grado di recuperare le tue perdite precedenti. I registri ti aiuteranno anche a tenere traccia di profitti/perdite a fini fiscali, se applicabile.

Un buon pianificatore

Ognuno ha la sua strategia (come il modo di fare le cose). Alcune persone tendono a fare vendite allo scoperto e a fare più vendite in un giorno. Altri colpiscono

la loro fortuna dopo molto tempo e guadagnano una grande quantità con una singola vendita.

Anche se effettuano una singola vendita in una settimana, possono guadagnare più di chi fa tante vendite in un giorno. Una volta trovata la strategia giusta per te stesso, devi attenerti a quella strategia. È fondamentale quale strategia scegli per te stesso. È perché nel trading di opzioni, la giusta strategia e tecnica per negoziare ti porterà in cima, e il male farà il contrario.

Consistenza

Nient'altro che piccoli pezzi possono essere guadagnati senza coerenza. Devi dare molto per ottenere molto. Nel caso del trading di opzioni, devi investire il tuo tempo per acquisire esperienza. Più esperienza acquisisci, più impari.

Come se impari quando e dove mettere i tuoi soldi; e quando tirarlo fuori. Molte persone tornano quando vedono guadagni minori, non sapendo che passi più piccoli portano a passi più grandi.

Imparare dai fallimenti

Proprio come la maggior parte degli uomini d'affari perde i propri soldi, allo stesso modo ogni trader subisce delle perdite. Questa è solo una parte del gioco! Ma un trader di successo non si arrende alla sua perdita e cerca di evitare la perdita in futuro dall'esperienza che ha acquisito dalla sua perdita precedente.

Poi arriva un momento in cui ha imparato ogni possibile ragione che ha portato alla sua perdita. In futuro, può coprire le sue perdite precedenti evitando lo stesso errore.

Interpretazione di notizie

I commercianti devono essere in grado di interpretare le notizie. Se sei bravo a interpretare le notizie, avrai le conoscenze esatte per prevedere quale sarà il prodotto che darà profitto e quando. Se puoi prevedere il futuro, sarai in grado di aumentare le tue entrate investendo in un determinato prodotto.

Sarai anche in grado di prevedere quando acquistare o vendere il prodotto per massimizzare il tuo profitto. Alcune notizie sono solo l'hype; devi essere in grado di distinguere tra le notizie reali e l'hype.

Non seguire cosa stanno facendo gli altri

Ognuno ha il suo pensiero. Ma una volta che entri nell'attività commerciale, non affidarti completamente alle strategie degli altri. Per essere un trader di successo, non devi fare quello che fanno tutti gli altri. Devi essere senza limiti ed essere te stesso. Potresti uscire dalla tua zona di comfort, ma seguire la tua passione sarà la chiave del successo. La tua disponibilità a correre dei rischi ti avvantaggerà nell'essere te stesso.

Pazienza

Essere un trader di opzioni di successo richiede molta pazienza. Devi essere in grado di aspettare fino a quando; è il momento giusto per eseguire l'azione. Significa che se devi mettere i tuoi soldi in qualcosa, devi aspettare finché non pensi; è il prezzo più basso per un determinato prodotto. Allo stesso modo, durante la vendita, devi essere paziente fino a quando non raggiungi la massima quantità di profitti.

Se non sei paziente durante il trading; una grande perdita potrebbe essere sulla tua strada!

Flessibilità

I tassi di mercato cambiano ogni giorno; devi essere in grado di imparare le mutevoli dimensioni del mercato. Devi conoscere le mutevoli tendenze del mercato e adottare nuove strategie.

Devi essere ben consapevole delle notizie rilevanti e credere sempre di essere uno studente. Devi accettare le perdite poiché la perdita in qualsiasi campo di lavoro è inevitabile. Devi accettare dovunque vada il mercato, che ti piaccia o no.

Gestione del rischio

Nelle opzioni, giochi in milioni, quindi c'è un rischio enorme. Devi gestire quanto rischio puoi sopportare in un determinato momento. Essere senza limiti non significa che devi dimenticare ciò che stai rischiando mentre metti i tuoi soldi. Se assegni un certo capitale a un investimento, potresti essere in grado di evitare un rischio maggiore di perdita ma, maggiore è il rischio, maggiori saranno i guadagni o le perdite.

Conclusione

Se possiedi queste determinate abilità o qualità, un giorno sarai il trader di maggior successo nel trading di opzioni. Sii paziente e sii coerente nel tuo lavoro. Le porte del successo continueranno ad aprirsi per te. Più

l'importante è credere in se stessi; se corri rischi maggiori e hai fiducia in te stesso, non c'è niente che tu non possa fare.

7. COME SELEZIONARE L'OPZIONE GIUSTA PER UN GRANDE GUADAGNO



Spartendo da semplici acquisti a spread più complicati come le farfalle, le opzioni hanno una miriade di strategie. Inoltre, sono disponibili per una gamma più ampia di valute, azioni e materie prime, contratti futures e fondi negoziati in borsa.

Spesso ci sono centinaia di scadenze e prezzi di esercizio per ogni opzione. Ciò rappresenta una sfida per i principianti nel selezionare l'opzione migliore tra molte. Ecco come puoi farlo come un professionista.

Cerca l'opzione giusta

Immagina di avere già in mente un asset su cui desideri fare trading, come una merce. L'hai scelto da uno screener di azioni attraverso la tua intuizione e l'intuizione di altri (ad esempio la ricerca). Indipendentemente dalla tecnica di selezione, dopo aver identificato il tuo asset per il trading, devi seguire questi passaggi per raggiungere il tuo obiettivo.

- Inquadra lo scopo dell'investimento
- Determina il tuo profitto
- Analizzare la volatilità
- Riconoscere gli eventi
- Crea una strategia
- Avvia i parametri delle opzioni

Questi passaggi seguono un processo logico, che semplifica la selezione dell'opzione giusta su cui fare trading. Analizziamo cosa rivelano questi passaggi.

Inquadra il tuo scopo

La base per entrare nel business del trading è trovare lo scopo dell'investimento. Perché vuoi iniziare a fare trading di opzioni? Qual è lo scopo dietro questo? Vuoi fare soldi veri o è solo un'attività secondaria? Poniti queste domande. Fai un quaderno e scrivi tutte le risposte che

hai.

Ora potresti pensare, perché così? È perché devi essere chiaro su questo punto. L'utilizzo di opzioni per fare soldi veri è molto diverso rispetto all'acquisto per speculazione o copertura.

Questo è il tuo primo passo e costituirà la base per altri passaggi. Quindi, allacciati!

Payoff del commerciante

Nella seconda fase, determina il tuo payoff di rischio e rendimento. Questo dovrebbe dipendere dal tuo appetito o dalla tua tolleranza al rischio. Conosci il tuo tipo: come se fossi uno dei trader conservatori, l'utilizzo di strategie aggressive per il trading di opzioni potrebbe non essere adatto a te. Queste strategie includono l'acquisto di opzioni esaurite in grandi quantità o la scrittura di put ecc.

Ogni strategia ha un profilo di rischio e rendimento ben fatto. Devi tenerlo d'occhio. E non dimenticare di valutare il tuo profitto alla fine della giornata!

Analizza la volatilità

È uno dei passaggi più cruciali nel trading di opzioni. Devi sicuramente analizzare la volatilità implicita. Confrontalo con la cronologia della volatilità delle azioni delle opzioni, più il livello di volatilità nel mercato.

Questo ti permette di conoscere il pensiero degli altri trader. Indipendentemente dal fatto che si aspettino che le azioni si muovano rapidamente o al rialzo in futuro o meno, se c'è un'elevata volatilità, ci sarà anche un premio più elevato. In questo caso, la scrittura delle opzioni sarà più adatta a te.

Un tasso più basso di volatilità implicita significa che ci sarà un premio più basso – buono per l'acquisto di opzioni (se pensi che le azioni si muoveranno di più e quindi anche il loro valore aumenterà).

Riconosci gli eventi

Esistono due tipi principali di eventi: specifici per il titolo e per tutto il mercato. Gli eventi specifici delle azioni includono tipi come spin-off, lanci di prodotti e rapporti sui guadagni, ecc. Gli eventi a livello di mercato, d'altra parte, sono quelli che hanno un ruolo enorme in ampi mercati come i rilasci di dati economici e la Federal Reserve

Annunci.

È importante conoscere e riconoscere ogni tipo di evento. Dal momento che hanno un enorme impatto sulla volatilità implicita e, quindi, possono avere un grande impatto sul prezzo quando si verifica. Riconoscere gli eventi prima che possano aiutare i trader a realizzare un profitto maggiore, determinare il momento giusto e la data di scadenza appropriata per la tua operazione.

Crea la tua strategia

I primi quattro passaggi ti consentono di vedere chiaramente la strada per la tua attività di trading di opzioni. Ora sei in una buona posizione per escogitare il tuo piano dopo aver conosciuto gli eventi, la volatilità implicita, il rischio e il rendimento della ricompensa e il tuo obiettivo di investimento.

Supponiamo che un trader conservatore con un buon portafoglio desideri guadagnare un premio entro alcuni mesi. Dovrebbe quindi utilizzare la tecnica di "scrittura" della chiamata coperta per raggiungere il suo obiettivo. Mentre un investitore aggressivo che prevede un calo del mercato entro alcuni mesi dovrebbe acquistare put sui titoli principali e così via.

Parametri di avvio

Dopo che il quinto passaggio è chiaro nella tua mente, prova ad avviare i parametri. Ad esempio, dovresti stabilire il tempo di scadenza, il delta dell'opzione e il prezzo di esercizio, ecc. Supponiamo che un trader voglia acquistare la chiamata con la data di scadenza più lunga. Ma vuole anche pagare un premio basso. In questo caso, la chiamata out-of-the-money è la più appropriata per lui. Ma per una chiamata ad alto delta, concentrati sull'opzione in-the-money.

In breve, segui i passaggi indicati per realizzare un buon profitto e affermarti come trader di opzioni professionale sul mercato. Determina il tuo obiettivo di investimento, analizza il rischio e la ricompensa, valuta la volatilità, pensa agli avvenimenti, crea la tua strategia e quindi personalizza i parametri delle opzioni.

Come guadagnare con il trading di opzioni

Un trader di opzioni guadagna denaro acquistando o vendendo o scrivendo opzioni.

Con il trading di opzioni, non solo acquisti o possiedi azioni di una società, ma sei anche in grado di vendere quelle azioni in futuro. Se conosci il diritto

strategie, puoi guadagnare oltre 100.815 \$ attraverso il trading di opzioni. Imparare le giuste strategie, conoscere i rischi, conoscere il mercato, moltiplicare il profitto e costruire ricchezza ti aiuterà a guadagnare di più con il trading di opzioni.

Guadagnare con il trading di opzioni nel 2020

Abbiamo arruolato alcune tecniche speciali di trading di opzioni che potrebbero aiutarti a capire e guadagnare meglio attraverso il trading di opzioni.

Ricapitolare

Prima di entrare davvero nel business, è necessario ricapitolare cos'è realmente il trading di opzioni, quali sono i suoi termini e come viene fatto con rischi minimi coinvolti. In parole povere, il trading di opzioni consiste nell'acquistare e vendere contratti di opzioni.

Il trading di opzioni non ti consente di votare o ricevere dividendi o qualsiasi altra cosa che un proprietario (parziale) della società possa fare. È solo un contratto tra te e un'altra parte che ti concede il diritto di acquistare (ad esempio "opzione call") o vendere (ad esempio, "opzione put") azioni di una società a un determinato prezzo.

È uno degli strumenti di "leva" più basilari a disposizione degli investitori che stanno cercando di aumentare il loro potenziale profitto accettando l'aumento del rischio che ne deriva sempre.

Ci sono alcuni termini chiave essenziali che vengono normalmente utilizzati nel trading di opzioni:

Azione

Un'opzione su azioni è un contratto tra la società e l'acquirente dell'opzione su azioni per acquistare o vendere 100 azioni della società a un determinato importo entro un determinato periodo di tempo.

Scadenza

Nel trading di opzioni, è coinvolto un contratto e ogni contratto ha una data di scadenza. Puoi acquistare o vendere le tue opzioni prima della data di scadenza, ma una volta superata la data di scadenza, il contratto non ha alcun valore. In tal caso, solo uno scrittore di opzioni può guadagnare.

Prezzo di sciopero

Un prezzo di esercizio è un prezzo al quale la merce o il bene deve essere acquistato o

venduto al momento dell'esercizio dell'opzione. Ad esempio, se il prezzo di esercizio è XYZ dollari per un'opzione call, è possibile esercitare il contratto acquistando l'azione identificata al prezzo di esercizio.

Premium

Premio Opzioni è il prezzo che deve essere pagato dal soggetto che acquista il diritto compra/il diritto di vendere, al soggetto che vende il diritto di comprare/il diritto di vendere, a titolo di premio per stipulare un contratto per il rischio dell'opzione esercitata se il contratto è in the money, che il venditore (venditore) dell'opzione sopporta durante la stipula del contratto di opzione. Dipende dal prezzo di esercizio, dalla volatilità del sottostante e dalla data di scadenza.

Opzione Call e Put

Nel trading di opzioni, c'è una call e un'opzione Put Una call è quando si acquista o si acquista un'azione e la put è quando si vende un'azione. Puoi acquistare o vendere un titolo prima della data di scadenza.

Asset sottostante

L'attività sottostante è un titolo di riferimento (azioni, obbligazioni, contratti futures, ecc.) su cui si basa il prezzo del titolo derivato come un'opzione. Ad esempio, le opzioni sono strumenti derivati, il che significa che i loro prezzi sono derivati dal prezzo di un altro titolo. Più specificamente, i prezzi delle opzioni sono derivati dal prezzo di un'azione sottostante.

Stile opzione

Un contratto di opzione è composto da due diversi stili; Stile americano o stile europeo. Le opzioni possono essere esercitate in un modo particolare ed entrambi gli stili ti consentono di esercitarle in modo diverso. Le opzioni in stile americano possono essere utilizzate in qualsiasi momento prima della scadenza, mentre le opzioni in stile europeo possono essere utilizzate solo alla data di scadenza stessa.

Moltiplicatore del contratto

Il moltiplicatore del contratto indica la quantità dell'attività sottostante che deve essere consegnata in caso di esercizio dell'opzione. Per le stock option, ogni contratto copre 100 azioni.

Valore relativo

Vendere una merce a un prezzo più alto rispetto al prezzo di acquisto e acquistare a un prezzo inferiore rispetto al mercato o come l'hai venduta.

Guadagnare con il trading di opzioni nel 2020

I trader di opzioni utilizzano diverse strategie per valutare l'operazione. Un elenco di strumenti è incluso nel processo di valutazione.

L'elenco potrebbe includere; analisi, storia, statistiche, stabilità, debito, dividendi, ecc.

Con una piccola lettura, un trader può facilmente ridurre al minimo il rischio di perdere il suo investimento. Ecco le 10 migliori strategie su come guadagnare con il trading di opzioni:

chiamata nuda

Una naked call è una strategia di opzioni in cui un investitore vende (un'opzione call) senza la sicurezza di possedere il titolo sottostante.

Put coperto

Un'opzione coperta è una strategia in cui il titolo viene acquistato o posseduto e viene venduta un'opzione. Il titolo sottostante viene acquistato e contemporaneamente si scrive o vende un'opzione call su quelle stesse azioni. Il Covered Put ha anche un profitto più alto nel caso in cui il titolo scenda al prezzo di esercizio delle short put.

Ad esempio, un investitore utilizza la sua opzione call (acquista) su un titolo che rappresenta 100 azioni per opzione call. Per ogni 100 azioni che l'investitore acquista, venderà un'opzione call contro di essa. Questa strategia viene definita chiamata coperta perché, nel caso in cui il prezzo di un'azione aumenti rapidamente, la chiamata corta di questo investitore è coperta dalla posizione lunga dell'azione.

La formula per calcolare il profitto massimo è:

Profitto massimo = Premio ricevuto - Commissioni pagate

Profitto massimo raggiunto quando il prezzo del sottostante \leq prezzo di esercizio dello short put

Sfruttare al massimo le opzioni

Le opzioni sono come un business; non tutti possono ottenere salari o redditi elevati. Per essere un uomo d'affari di successo, hai bisogno di un certo tipo di mentalità, poche abilità e un po' di capitale. Mentre discutiamo di come fare soldi attraverso le opzioni, non dobbiamo guardare solo a strategie fisiche ma anche a strategie mentali e indirette.

Strategie indirette:

Conservazione dei registri

Tenere un registro corretto dei tuoi progressi è davvero utile per una vita pratica di successo così come per gli affari. Nelle opzioni, puoi monitorare i tuoi progressi, i punti deboli e le ragioni di questi punti deboli. Ti aiuterà a imparare dai tuoi errori ed evitarli in futuro.

Stai attento

La tecnologia continua a svilupparsi ogni giorno; ogni giorno vediamo nuove innovazioni. Per camminare con gli altri ed evitare di stargli dietro, devi avere accesso alle ultime notizie e agli aggiornamenti sulla tecnologia ed essere pronto per quello che verrà dopo.

Rimani aggiornato

Per essere un trader di opzioni di successo, devi essere aggiornato su cosa sta succedendo nel mercato azionario e su come e quando i prezzi cambieranno. In questo modo sarai in grado di prevedere i prezzi di mercato e pianificare la tua mossa di conseguenza per evitare perdite

Strategie mentali:

Gestire il rischio

È un famoso detto: "Taglia il tuo cappotto secondo la stoffa". Applicandolo alle opzioni, devi accettare di non investire tutti i tuoi soldi in un unico prodotto. Investi solo l'importo di cui puoi sopportare il rischio.

Gestire il tempo

Nessuno può obbligarti a mettere o chiamare i soldi in un determinato prodotto finché non lo desideri tu stesso. Cerca il momento perfetto per farlo, investi solo quando sai che è un momento perfetto. La cosa più importante è la pazienza. Ma tieni traccia della data di scadenza.

Separare la vita pratica da quella lavorativa

Per fare progressi come trader di opzioni, non confondere mai le tue emozioni con la tua attività. Se stai attraversando qualcosa di brutto nella vita reale e ti senti frustrato, prenditi una pausa. Una mente fresca può pensare bene della mente impegnata a risolvere altri problemi. Il trading di opzioni è il gioco della mente, quindi prenditi una pausa e torna quando sei rilassato.

Queste sono le poche strategie fisiche, mentali e indirette che abbiamo discusso sopra. Spero che questo ti porterà alle vette del successo e ti renderà

la maggior parte delle opzioni.

8. CONSIGLI E TRUCCHI IMPORTANTI SUL TRADING DI OPZIONI

UN Il trading di opzioni è una parte del trading che ti consente di negoziare le tue aspettative di mercato controllando anche il rischio con cui parteciperai a questo trading. Ora, se hai un'idea più chiara di come eseguire correttamente il trading di opzioni, non ci sono limitazioni. Ciò significa che puoi negoziare varie strategie e cercare profitto in tutti i tipi di condizioni di mercato.

Tuttavia, questo trading di opzioni non significa che devi scambiare le strategie delle tue opzioni di trading complesse per trarne profitto. Invece, puoi spendere i tuoi soldi in modo più efficace per ottenere profitti semplicemente sostituendo le tue normali posizioni di trading con l'aiuto delle opzioni.

Un piccolo approfondimento:

Con l'inizio del 2020, l'attività di trading di opzioni ha raggiunto un drastico aumento. Ora, se calcoliamo l'aumento dei contratti di opzione in quest'anno fino ad ora, è stata calcolata una stima di un aumento del 53% - rispetto a quello dello stesso periodo dell'anno scorso. Quindi, non può esserci certamente un momento migliore per dedicarsi all'attività di trading sulle opzioni, se ci stai pensando.

Tuttavia, è molto importante comprendere le strategie delle opzioni di trading e come possono essere eseguite correttamente. Pertanto, anche se sei un professionista nel trading, è importante conoscere i principali e importanti suggerimenti relativi al trading di opzioni. Ora, queste strategie e questi suggerimenti possono cambiare in base alle condizioni e ai criteri del mercato.

Ma per darti una risposta coerente su come eseguire con fermezza il trading di opzioni, discuteremo di seguito alcuni suggerimenti che potrebbero fare al caso tuo. Quindi, senza ulteriori indugi, andiamo avanti e scopriamo alcuni di questi suggerimenti!

Segui un piano di uscita ben definito

Controllare le tue emozioni mentre fai trading con le opzioni può essere cruciale per aiutarti a ottenere grandi profitti. Questo passaggio cruciale può essere definito semplicemente

avere un piano per lavorare e attenersi sempre ad esso. In questo modo, sei ben consapevole del risultato che desideri quando segui quel piano e puoi sicuramente raggiungerlo. Quindi non importa quanto le tue emozioni ti costringano a cambiare idea e dimenticare il nostro piano mentre ci sei, assicurati di non farlo!

Ora, quando fai un piano ben definito, non puoi perderti la pianificazione delle uscite anche qui. Questo piano di uscita non significa che dovresti ridurre al minimo la tua perdita in termini di fronte al lato negativo del trading di opzioni. Ma invece, avere un piano di uscita ben definito e un piano di uscita al ribasso in anticipo può aiutarti a uscire dall'operazione al momento giusto, anche se l'operazione sta andando bene secondo il tuo piano.

Questo è molto importante perché il trading di opzioni è un'attività che affronta un decadimento dei tassi quando la data di scadenza inizia ad avvicinarsi.

Educa te stesso

Le opzioni di trading possono essere un'attività complessa rispetto al semplice acquisto e vendita di azioni. E se non capisci bene questa attività, ci sono possibilità che tu non sia in grado di arrivare da nessuna parte. Tuttavia, se continui a cercare conoscenza ed esperienza in questo, sarai meglio consapevole di come puoi investire qui e ottenere profitti.

Ora, per iniziare, devi avere una valutazione adeguata del tuo piano di investimento. Questa valutazione può includere i tuoi obiettivi individuali, i vincoli di rischio, l'orizzonte temporale, i vincoli fiscali e le esigenze di liquidità che hai.

Non raddoppiare per le perdite passate

Se stai pensando di raddoppiare su una strategia di opzioni solo perché vuoi coprire la tua perdita passata, allora sicuramente non andrai lontano con questo. Una semplice ragione per questo è che le opzioni nel trading di opzioni sono semplicemente derivati e le loro proprietà di prezzo non sono le stesse delle azioni sottostanti.

Pertanto, anche se a volte ha senso raddoppiare solo in modo da poter recuperare la perdita che hai subito in precedenza (e perché lo segui nelle azioni), non significa che ti servirà anche con profitto quando sei nella galassia delle opzioni.

Quindi, invece di aumentare il rischio, dovresti semplicemente fare un passo indietro e chiudere l'operazione. In questo modo, puoi ridurre più perdite che potrebbero essere ulteriormente

vieni nello stesso mestiere e scegli semplicemente un'altra opportunità. Di conseguenza, invece di scavare più a fondo nella categoria delle opzioni specifiche, accetterai la tua perdita e ti salverai da una caduta più grande.

Gestisci il tuo rischio

Ora l'aspetto più importante qui è il rischio del trading di opzioni. Quindi, quando vai per il trading di opzioni, devi capire quanti rischi puoi correre. Che tu sia un principiante o qualcuno che è stato nel trading di opzioni per un po', avere una certa valutazione del rischio che puoi gestire è molto importante.

Una volta che lo hai, puoi esaminare i diversi metodi che possono aiutarti a gestire i tuoi rischi. Ora per gestire i rischi, puoi scegliere diverse opzioni per tutta la durata del contratto di opzioni: gestire i rischi. Queste diverse opzioni includono:

Chiusura di un'operazione: si riferisce principalmente all'assunzione di una posizione di compensazione nell'operazione. Quindi, se hai acquistato un'opzione call nelle opzioni di trading, puoi semplicemente impostare l'opzione call e chiudere l'operazione per gestire il rischio in tempo.

Consentire la scadenza di un'opzione

Questo può essere possibile quando un contratto in opzioni di trading ha raggiunto la sua data di scadenza senza essere lavorato. Qui puoi anche acquistare o vendere una call o una put, secondo il contratto.

Lancia un'opzione

Questo è principalmente il processo di gestione del rischio semplicemente chiudendo un'opzione che è vicina alla data di scadenza e quindi investendo contemporaneamente in una categoria simile di commercio che ha una data di scadenza lontana.

Incarico

Infine, un'altra strategia per gestire i rischi nelle opzioni di trading è semplicemente andare per un incarico. Questo è possibile quando vendi un'opzione semplicemente ricevendo o consegnando le azioni che si trovano sotto lo stock di quell'opzione.

Finalmente

Ora le opzioni di trading sono un aspetto di trading abbastanza familiare per molti, ma la maggior parte dei nuovi trader non ne ha molta familiarità. Tuttavia, ottenere grandi profitti e successo nel trading di opzioni è qualcosa che chiunque può ottenere. Soltanto

se ti istruisci in questo, acquisisci un po' di esperienza e segui correttamente suggerimenti e trucchi efficienti (come menzionato sopra), sei sicuro di andare lontano nelle opzioni di trading.

9. FAQ IMPORTANTI SUL TRADING DI OPZIONI

W Abbiamo cercato di includere tutte le domande frequenti di base e importanti relative al trading di opzioni. Spero che ti aiutino a capire meglio il trading di opzioni e in caso contrario. Puoi inviare le tue richieste nella sezione commenti.

Esiste una definizione di opzioni?

Le opzioni sono derivati supportati dal valore dei titoli sottostanti come le azioni.

Le opzioni stanno mettendo i tuoi soldi per il diritto di acquistare un'azione a un prezzo specifico prima della sua data di scadenza. Ci sono due tipi di opzioni; opzioni comprare o vendere.

Quando un investitore prende parte alle opzioni, sta comprando o vendendo un contratto di opzioni e scommettendo che l'azione sottostante aumenterà o diminuirà di prezzo prima della data di scadenza.

Come ottenere il massimo profitto nel trading di opzioni?

Secondo Allen Everhart, il modo migliore per massimizzare i profitti nel trading di opzioni è semplicemente mantenerlo semplice. Nelle sue parole, "Ho imparato ad apprezzare l'acquisto di opzioni call deep-in-the-money/deep-in-time nonostante il denigrazione che questa strategia ottiene".

Acquista una chiamata delta 70 se pensi che il mercato stia andando al rialzo o put se pensi che il mercato stia scendendo. Non dovrai preoccuparti molto del decadimento theta (c'è poco, ma non molto) e guadagnerai l'80% di una mossa di \$ 1 sul titolo o sull'ETF a un costo molto inferiore rispetto a un numero equivalente di azioni di azioni, e non c'è il rischio di essere esercitati casualmente e che il titolo (lungo o corto) appaia improvvisamente nel tuo account la mattina successiva!

Quando hai 200 posizioni short su opzioni e una dozzina di esse vengono esercitate casualmente durante la notte, apprezzerai l'approccio "semplice" al trading di opzioni.

Cosa succede in caso di scadenza del contratto di opzioni?

Nel caso in cui il contratto raggiunga la sua data di scadenza e tu non abbia ancora esercitato il tuo diritto alle opzioni, perderai il tuo diritto e il premio. Il contratto diventa nullo.

L'unica persona che ne trarrà profitto è l'autore del contratto.

Qual è la differenza tra prezzo di esercizio e prezzo delle azioni?

Un prezzo di esercizio è un prezzo al quale il titolare di un'opzione può eseguire il contratto, mentre; un prezzo di un'azione è l'ultimo prezzo di transazione di almeno una singola azione di un sottostante.

Cos'è la chiamata nuda?

È una strategia in cui un investitore scrive un'opzione call senza avere una posizione nel titolo sottostante stesso. Per impostare una naked call, un investitore vende semplicemente un'opzione call senza possedere il titolo sottostante. Se scrive una semplice chiamata e il titolo sale del 100 o 200%, lo scrittore deve consegnare, ma è una strategia ad alto rischio.

Cos'è il contratto americano?

Un'opzione americana è una versione di un contratto di opzione che consente ai titolari di esercitare i diritti di opzione in qualsiasi momento prima e compreso il giorno di scadenza.

Che cos'è un contratto europeo?

Un contratto europeo ti consente di esercitare solo il giorno della scadenza.

Quale è il contratto europeo più redditizio o americano? contrarre?

Un'opzione contrattuale americana consente all'investitore di esercitare in qualsiasi momento prima della data di scadenza mentre, in opzioni contrattuali europee durante il loro "periodo di esercizio" (di solito giusto quando scadono, ma non prima).

Quindi un contratto all'americana si esercita di più ma è di più? redditizio?

Se esercizio il mio contratto americano prima della sua data di scadenza, l'investitore potrebbe ottenere maggiori profitti, ma potrebbe anche perdere denaro. Matematicamente, non c'è vantaggio, poiché un investitore può realizzare la stessa quantità di profitto

esercitando il suo diritto il giorno della sua scadenza.

Quando dovresti iniziare a fare trading di opzioni?

Dovresti iniziare a fare trading di opzioni quando hai abbastanza investimenti e risparmi. Hai bisogno delle conoscenze, dei dati e delle strategie adeguate sul mercato. Dovresti essere in grado non solo di prevedere, ma anche di implementare le strategie al momento giusto.

Quanto dovresti investire nel trading di opzioni?

Si consiglia di iniziare il tuo investimento con 5.000\$ a 15.000\$. Cerca di non investire tutti i tuoi risparmi o le tue entrate poiché c'è un fattore ad alto rischio coinvolto nel trading di opzioni.

Quando dovresti esercitare le tue opzioni?

Secondo Bill Bischoff, dovresti esercitare le tue opzioni all'ultimo minuto.

L'ultimo minuto è quando lo stock è salito al punto in cui sei pronto per lo scarico, o appena prima della data di scadenza dell'opzione, a seconda di quale evento si verifica per primo. All'ultimo minuto o alla data di scadenza, sai che non c'è andare più in alto di questo, quindi puoi facilmente esercitare le tue opzioni, anche se, nell'ultimo giorno, il costo delle tasse è solitamente più alto.

La migliore strategia di trading di opzioni?

Secondo Allen Everhart, non esiste una cosa del genere è la migliore strategia. Quotidiano stock o tasso di mercato è diverso. Anche l'attività o le società sottostanti sono diverse l'una dall'altra. Non puoi applicare una strategia a tutte le tue opzioni. Tuttavia, tutte le strategie di trading di opzioni sono direzionali.

Che cos'è una strategia short put e long call?

Una put corta e una chiamata lunga sono la stessa cosa. Il lancio del peso e la chiamata lunga fanno soldi quando il titolo sale. Ma lo short put è noto per essere un po' più rischioso della strategia long call.

La put corta può essere esercitata se il titolo non scende, e in tal caso è possibile mantenere il premio dell'opzione.

Come guadagna uno scrittore di opzioni?

Uno scrittore di opzioni guadagna quando l'azione o il premio che è stato acquistato raggiunge la sua data di scadenza senza essere esercitato. In tal caso, l'autore dell'opzione mantiene l'intero premio.

In che modo gli investitori perdono denaro nel trading di opzioni?

Non c'è un motivo specifico per cui un investitore perde i suoi soldi. Ci sono diversi casi e scenari. Ma gli errori più comuni che le persone commettono sono non raccogliere informazioni sufficienti o la loro mancanza di conoscenza. La maggior parte delle persone pensa che sia un modo breve per diventare ricchi o può credere troppo nella "fortuna".

CONCLUSIONE

T grazie per aver scaricato questo libro. L'obiettivo di scriverlo era far capire ai dilettanti, ai principianti e persino ai professionisti i concetti difficili e talvolta difficili da digerire.

Il linguaggio di questo libro è, quindi, semplice, facile e facile da usare (nel senso che chiunque può coglierne il significato). Inoltre, abbiamo aggiunto quanti più esempi possibile con ogni nuovo concetto in modo che il lettore non si confonda.

Alla fine, finiremmo da dove siamo partiti: "imparare i concetti del trading di opzioni potrebbe sembrare difficile, ma una volta che li prendi, sono tuoi!"

Quindi, ricorda, devi essere paziente, tollerante al rischio e un pianificatore consapevole quando si tratta di affari. Buon commercio. Possa ciascuno dei tuoi investimenti darti più profitti di quanto ti aspettassi.

GIORNO E ALTALENA COMMERCIO

Guida per principianti.

Strategie di investimento in azioni, opzioni e forex. Aggiungere entrate extra e ottieni la tua libertà finanziaria

INTRODUZIONE

io. In questo libro parleremo di due tipi di trading: day trading e swing trading. In questa versione estesa abbiamo aggiunto una parte sul trading di opzioni. È necessario che ogni principiante comprenda tutte le sezioni.

Per iniziare la tua carriera come trader di successo, dovresti avere alcune conoscenze di base sul trading. Questo libro ti aiuterà a chiarire i tuoi dubbi riguardo al day, swing e trading di opzioni, come dovrebbero investire i loro soldi e quale commercio darà loro più profitto.

Il day trading è definito come l'attività di acquisizione di profitto dalla variazione dei prezzi delle azioni durante il giorno. In altre parole, il day trading riguarda la vendita e l'acquisto di azioni in breve tempo.

D'altra parte, lo swing trading è definito come l'attività di acquisto e vendita di azioni durante un arco di tempo di più di un giorno.

Per "Trading" si intende l'insieme delle azioni volte a realizzare un reddito extra attraverso strategie di investimento. Coloro che applicano la strategia di trading nel mercato finanziario per ottenere profitti e mantenere una posizione nel mercato sono conosciuti come "Swing Traders". Gli swing trader in genere si concentrano sull'ottenere profitti con piccoli guadagni in un lasso di tempo molto breve. La loro posizione nel mercato può essere mantenuta per diversi giorni o per una settimana. Comprano alcune azioni sul mercato e poi le vendono quando credono di poter ottenere un buon profitto su di esse. È un processo in corso tra un acquirente e un venditore. Gli swing trader mantengono il controllo e l'equilibrio sul mercato; quindi, con l'aiuto dell'analisi tecnica, identificano le oscillazioni dei prezzi e determinano se acquistare o meno un'azione, se i prezzi stanno aumentando o se possono uscire con una perdita. Il rischio dei trader swing è 0,5% su ogni operazione; tuttavia perseguono obiettivi molto più elevati in termini di profitto da ogni commercio. La maggior parte dei trader acquista le azioni quando i prezzi sono bassi e le vende quando i prezzi sono più alti sul mercato. In alcuni casi, questa strategia provoca perdite, perché i valori delle azioni diminuiscono (questo accade in casi molto rari, però).

La maggior parte delle persone fa confusione tra lo swing trading e il day trading. In realtà, lo swing trading è molto diverso dal day trading perché il day trading ha minori possibilità di ottenere perdite rispetto allo swing trading. Inoltre, nel day trading, il tempo di negoziazione è limitato (un giorno), mentre nello swing trading non c'è limite di tempo: si tratta infatti di un processo notturno. Il mercato degli swing traders lavora per 24 ore; pertanto, le possibilità di profitto/perdita sono maggiori perché il venditore può affrontare la caduta in qualsiasi momento. Gli swing trader sono anche ricompensati da una leva del 50%. Ad esempio, se a un trader è consentito avere un margine nel suo

trading, allora ha solo bisogno di investire \$ 20.000 per uno scambio piuttosto che investire \$ 40.000. È un processo attraverso il quale è possibile detenere profitto per lungo tempo. Questo processo non riguarda l'ottenimento di profitti in un giorno, ma può durare da un giorno a molti giorni. Ci saranno più possibilità di ottenere un profitto che una perdita. Per i principianti, lo swing trading è l'opzione migliore da scegliere perché lo swing trading è facile e ci sono più possibilità di successo. È, infatti, meno stressante per un venditore ed è associato a maggiori probabilità di successo rispetto al day trading, poiché il tempo consente di migliorare l'accuratezza nella decisione di investimento. È risaputo che gli swing trader hanno diverse opportunità e opzioni per fare trading. Possono avere maggiori opportunità di investire i loro soldi durante il periodo di investimento più lungo. È molto importante per un trader preoccuparsi della variazione di prezzo delle proprie azioni ed essere paziente. Dopotutto, tutto ruota intorno a profitti e perdite.

Finora, tutti i siti Web o i libri sul day e swing trading che ho letto non erano adatti a una chiara comprensione di come entrare o uscire in qualsiasi sistema di trading. Ho scritto questo libro in modo che possa aiutare le prossime generazioni a capire di cosa tratta il day e lo swing trading e ad investire i loro soldi sul tipo di trading che più si adatta a loro.

Le opzioni sono una particolare tipologia di strumento derivato che dà la possibilità di acquistare o vendere, ad un prezzo prefissato, uno specifico sottostante, che rappresenta l'oggetto del contratto di opzione.

Il termine "opzione" deriva dalla parola possibilità. Le opzioni offrono l'opportunità di acquistare o vendere un bene. Rappresenta quindi un diritto per l'abbonato e non un obbligo.

I termini call, put, strike price, at the money e in the money sono tutti collegati alle operazioni di gestione delle opzioni. Il loro significato, come vedremo, è molto più intuitivo e immediato di quanto possa sembrare.

La recente evoluzione dei mercati finanziari ha dato grande risalto alle opzioni e alle loro particolari varianti come le opzioni binarie. Sono considerati strumenti finanziari redditizi e versatili per le strategie di trading. Offriranno a molti risparmiatori l'opportunità di costruire portafogli pay-off attraverso l'aggregazione di diversi strumenti.

1. DIFFERENZA TRA SWING TRADING E DAY TRADING

io Se sei un principiante, allora tieni la mente aperta per entrambi i tipi di trading, non sprecare soldi o tempo nell'investire finché non hai una conoscenza sufficiente dei sistemi di trading. Per un principiante, ci sono sempre molte cose da imparare, qualunque cosa il campo stia imparando sul tuo interesse è molto importante. Prima di iniziare qualsiasi modello di trading, è necessario comprendere le tue esigenze e aspettative che hai in relazione al sistema di trading. Prima di tutto, il trader che vuole fare trading dovrebbe sapere quanto trading attivo desidera. Se vuole un sistema di trading che sopravviva nel mercato per un tempo più lungo, oppure no. Quali sono le sue attuali esigenze e aspettative relative al trading se può gestire il suo sistema di trading per lungo tempo, o aveva abbastanza resistenza per sopportare la perdita? Una volta che hai deciso il percorso giusto che fa per te e che ritieni possa soddisfare le tue aspettative, puoi facilmente sopravvivere nel mercato del trading.

I trader hanno diviso il trading in due parti swing trading e day trading. Questo libro ti aiuterà a capire la differenza tra i due sistemi di trading e ti aiuterà nella scelta del sistema di trading migliore per te. L'obiettivo principale di qualsiasi principiante è quello di ottenere profitti dalle proprie azioni, che si tratti di un day trader o di uno swing trader; alla fine della giornata, ogni trader ha bisogno del proprio nome in una lista di profitto. Anche se entrambi i trading hanno strumenti e procedure di analisi tecnica diversi.

Il day trading consiste nel fare più operazioni in un giorno. Come suggerisce il nome, fare trading sulle tue azioni entro un giorno ma più volte è il day trading. Questo commercio consiste nel fare trading entro un giorno o ore. Non puoi andare oltre quel tempo, e non importa quante volte fai trading in un giorno. L'obiettivo principale dei day trader è fare soldi dal loro reddito preesistente in un giorno con le loro azioni. Questo commercio riguarda solo il day trading; non contiene ore notturne. Non detengono titoli overnight. Il più grande vantaggio del day trading è che il trader ha meno possibilità di subire qualsiasi tipo di perdita. Ad esempio, se un trader investe i propri soldi su qualsiasi azione e il tasso di mercato di tale azione scende nel

nel cuore della notte, quindi in tal caso, il trader deve affrontare una perdita, ma a causa del day trading, anche le possibilità di perdite sono inferiori in quanto non investiranno i propri soldi per la notte.

Secondo la Securities and Exchange Commission (SEC) degli Stati Uniti, "I day trader in genere subiscono perdite finanziarie nei loro primi mesi di negoziazione e molti non raggiungono mai lo stato di profitto". Dopo che la SEC ha messo in guardia il day trader, è stato facile capire che i trader principianti non dovrebbero correre rischi investendo denaro più di quanto possono permettersi, non dovrebbero andare oltre i loro limiti. La maggior parte delle persone si suicidano quando subiscono una perdita perché hanno preso in prestito denaro dai loro amici e familiari o da altre fonti.

I commercianti giornalieri non hanno bisogno di alcun partner, di solito lavorano da soli, non hanno orari flessibili e svolgono il loro lavoro in base al loro umore e alle loro esigenze. Di solito lavorano al loro posto e si tolgono e rappano tutte le loro cose ogni volta che vogliono. Non hanno bisogno dell'istruzione di nessuno perché svolgono il loro lavoro in modo indipendente.

A volte diventa difficile per i principianti dei day trader competere nel mercato perché, oltre a guadagnare e posizionarsi nel mercato, devono anche competere con i trader ad alta frequenza, mentre altri hanno maggiori vantaggi rispetto ai principianti quando diventano professionisti nel loro lavoro e hanno più esperienza di loro. Una volta che inizi a ottenere profitti dalle tue azioni, non ci sarà ritorno da questa avventura di guadagno; desidererai investire di più e guadagnare di più.

Il day trader deve generare un grande sforzo e usare le sue abilità per mantenere la posizione nel mercato. Un principiante che vuole avere tutti i lussi prima ha dovuto lasciare il suo lavoro per mantenere tutta la sua attenzione sul trading perché non sarebbe facile per un day trader continuare il suo lavoro poiché tutto dipende da te per mantenere il controllo e l'equilibrio nel mercato. Perché il day trading è stressante e il trader deve tenersi aggiornato sulla situazione in corso nel mercato. Dovrebbe essere consapevole dello screening multiplo, quindi lo aiuterà a individuare le opportunità di trading.

Per esempio

Se qualcuno continua il suo lavoro con il day trading e d'altra parte le azioni di un'organizzazione come Walmart (WMT) o Apple (AAPL) all'interno del mercato forex stanno salendo, e c'è un'eccellente opportunità per lui di scambiare la sua valuta come l'euro o dollaro USA (EUR/USD), quindi gli mancherà il

prospettiva a causa della sua routine lavorativa.

Quindi sarà meglio per il futuro del trader concentrarsi solo su una cosa alla volta, il suo lavoro o un business. Per il trading ci sono vari mercati; gli alti e bassi del mercato facilitano chi non può permetterselo. Una volta che le azioni scendono, gli acquirenti le prendono in prestito e le vendono quando il tasso sale. Nel mercato forex, è più facile per i principianti investire i propri soldi in quanto richiede il minor capitale per scopi di trading. Puoi iniziare da una piccola somma come \$ 50, ma se puoi investire una grande quantità sarà più utile. Per ottenere una buona posizione nel mercato, il commercio di azioni richiede almeno \$ 25.000 per effettuare scambi giornalieri. Le azioni giornaliere richiedono più capitale per la posizione migliore, ma se non hai \$ 25.000 o non puoi mantenere il tuo account sopra \$ 25, 000 quindi le azioni non sono il posto giusto per investire denaro o non sprecare il tuo tempo su di esso, ma se hai superato \$ 25.000, allora le azioni sono un mercato valido per il trading giornaliero. I commercianti giornalieri di lunga data amano le gioie di mettere il loro ingegno contro il mercato e altri professionisti giorno dopo giorno. La scarica di adrenalina del trading a raffica sono alcune cose che non molti trader ammetteranno, ma è un gigante pensare alla loro decisione di guadagnarsi da vivere con il day trading. È dubbio che questi tipi di persone si accontenteranno di trascorrere le loro giornate vendendo widget o leggendo i numeri in un cubicolo dell'ufficio. La scarica di adrenalina del trading a raffica sono alcune cose che non molti trader ammetteranno, ma è un gigante pensare alla loro decisione di guadagnarsi da vivere con il day trading. È dubbio che questi tipi di persone si accontenteranno di trascorrere le loro giornate vendendo widget o leggendo i numeri in un cubicolo dell'ufficio. La scarica di adrenalina del trading a raffica sono alcune cose che non molti trader ammetteranno, ma è un gigante pensare alla loro decisione di guadagnarsi da vivere con il day trading. È dubbio che questi tipi di persone si accontenteranno di trascorrere le loro giornate vendendo widget o leggendo i numeri in un cubicolo dell'ufficio.

Il day trading e lo swing trading presentano vantaggi e svantaggi. Nessuna delle due strategie è più salutare dell'opposto e i trader dovrebbero scegliere l'approccio che funziona meglio per le proprie capacità, preferenze e stile di vita. Il day trading è più adatto alle persone che sono colpite dal trading a tempo pieno e possiedono le tre D: risolutezza, disciplina e diligenza (prerequisiti per il day trading di successo). Lo swing trading, d'altra parte, non richiede un insieme di tratti così formidabile. Dal momento che lo swing trading sarà intrapreso da chiunque abbia un certo capitale di investimento e non richiede un'attenzione a tempo pieno, è un'opzione praticabile per i trader che vogliono mantenere il loro lavoro a tempo pieno, ma anche dilettersi nei mercati. Gli swing trader dovrebbero persino essere in grado di applicare una combinazione di analisi fondamentale e tecnica,

Se distinguiamo lo swing e il day trading, allora entrambi sono molto diversi l'uno dall'altro, il loro tempo di inquadramento per il commercio è diverso. Lo swing trading riguarda la vendita e l'acquisto di azioni per giorni e settimane. Gli swing trader hanno più tempo per fare trading rispetto ai day trader. Il loro frame di trading è più lungo dei day trader poiché mantengono una posizione durante la notte. A causa delle loro 24 ore di trading,

non possono evitare il rischio che può causare loro un grosso problema. Ci saranno più possibilità di perdere denaro in questo commercio. Devono preoccuparsi continuamente dello stock perché potrebbe essere diverso durante l'apertura o potrebbe essere diverso da come è stato chiuso prima del giorno. I trader swing hanno bisogno di molta pazienza perché ha dovuto affrontare molti problemi. Le operazioni in genere hanno bisogno di tempo per capire. Mantenere aperta una transazione per un asset per alcuni giorni o settimane può portare a profitti più elevati rispetto al trading in entrata e in uscita dallo stesso titolo più volte al giorno. Gli swing trader sanno che ci vorrà molto tempo, ma generalmente non lo fanno sembrare un lavoro a tempo pieno. Il commercio swing potrebbe richiedere alcuni giorni o forse alcune settimane per funzionare. Gli swing trader non fanno di questo trading una carriera lavorativa a tempo pieno come i day trader. Se hai abbastanza conoscenze sullo swing trading e sul capitale di investimento, puoi provare lo swing trading. Se sei un principiante nel trading e vuoi investire i tuoi soldi, allora lo swing trading è una scelta perfetta perché non richiede la tua attenzione 24 ore su 24, 7 giorni su 7, o non hai bisogno di incollarti davanti ai tuoi computer e mantenere il tuo occhi e dita incrociate tutto il tempo. Uno swing trader può anche fare un lavoro a tempo pieno se lo desidera, perché lo swing trading non richiede il controllo sugli schermi tutto il tempo. I requisiti di margine nello swing trading sono più alti e, di solito, la sua leva massima è due volte il proprio capitale, mentre il day trading ha quattro volte e il proprio capitale. Non ha bisogno di un monitoraggio costante come il day trading; può essere interrotto in qualsiasi momento ogni volta che ritieni che ci sia un rischio nell'esecuzione del processo di trading.

Come qualsiasi altro scambio di scambi, anche lo swing trading può comportare perdite sostanziali. Gli swing trader infatti devono affrontare una perdita maggiore rispetto ai day trader poiché gli swing trader mantengono la loro posizione sul mercato per un tempo più lungo. Ecco perché corrono il rischio di perdite maggiori rispetto al day trading. Gli swing trader di solito hanno il loro lavoro regolare e hanno anche altre fonti di reddito da cui possono superare le loro perdite. Ci sono più possibilità di burnout per gli swing trader a causa dello stress nello swing trading poiché raramente è un lavoro a tempo pieno. Gli swing trader hanno più possibilità di mitigare o compensare le loro perdite.

Lo swing trading può essere fatto avendo un solo computer e con gli strumenti di trading convenzionali a differenza del day trading non richiede la tecnologia dell'arte. Gli swing trader hanno una leva overnight del 50% rispetto al day trading, ma anche questo margine può essere rischioso, in particolare se si verificano richieste di margine. Questi scambi non riguardano tanto ciò che si desidera negoziare, che si tratti di materie prime, ad esempio futures sul petrolio o azioni del CAC 40. Invece, si tratta semplicemente di tempismo. Quindi, dove ci sono volute 4 ore e grafici giornalieri del giorno

trading, sarà più preoccupato per lo swing trading dove ci sono voluti grafici di più giorni e modelli di candele. I crossover a media mobile, i modelli Testa e spalle, le stelle cadenti, ecc. sono alcuni dei più popolari.

Lo swing trading può essere estremamente impegnativo in due mercati, l'ambiente del mercato ribassista o il mercato rialzista furioso. Qui vedrai che anche i trader molto attivi non verranno visualizzati nella stessa posizione, ci saranno le stesse oscillazioni al rialzo e al ribasso. Per investire in borsa è obbligatorio disporre di un metodo di trading ben organizzato. È molto importante mantenere le cose semplici, poiché nelle fasi iniziali sembrerà un po' difficile per i principianti, ma invece di farsi prendere dal panico, dovrebbero gestirle con sicurezza. Una volta imparate le regole dello swing trading e la disciplina, guadagnerai in grande quantità nel mercato azionario. Lo swing trading ti consente di fare trading in ogni situazione, che sia rialzista, ribassista o semplicemente di lato.

Gli swing trader utilizzano indicatori di analisi tecnica per identificare le oscillazioni dei prezzi nel mercato e determinare le condizioni del mercato, se il prezzo di un'azione scenderà o aumenterà nel breve periodo. Attraverso questo, investono il capitale in titoli che hanno slancio e selezionano il momento migliore per acquistare o vendere il titolo. Questi indicatori di analisi tecnica aiutano i trader a utilizzare i grafici swing per il loro swing trading sull'andamento della situazione attuale della sicurezza. Per analizzare l'attuale modello di trading, gli swing trader utilizzano grafici di swing trading, che aiutano il trader a fornire dati basati sull'analisi statistica. A differenza del day trading, lo swing trading non riguarda il valore a lungo termine del titolo; invece di questo, sono solo preoccupati per gli alti e bassi del prezzo delle azioni.

Lo swing trading e il day trading appaiono simili in alcuni aspetti del trading. Il fattore principale del trading è distinguere le due tecniche e mantenere la posizione in tempo sul mercato. A differenza dei day trader, non si chiude in pochi minuti o in poche ore, ci vogliono diverse settimane e giorni notturni. Sono soggetti all'imprevedibilità dei rischi durante la notte che possono comportare un significativo slancio dei prezzi. Gli swing trader possono controllare periodicamente le loro posizioni sul mercato e agire quando vengono raggiunti punti critici.

Le principali differenze tra lo swing trading e il day trading sono:

Orari di negoziazione:

Entrambi hanno tempi di negoziazione diversi. Nel day trading, sono necessarie da due a quattro ore al giorno per scopi di trading, e in questo lasso di tempo il trader riesce ad analizzare i grafici, entrare e uscire dalle posizioni e rivedere diversi titoli. Considerando che il minimo dello swing trader ha bisogno di 45 minuti al giorno, aggiorna il suo ordine e trova quello nuovo. Il day trading richiede più tempo dello swing trading.

Rischi:

Il day trader subisce più perdite rispetto agli swing trader perché i day trader potrebbero dover eseguire sei operazioni al giorno, mentre gli swing trader potrebbero dover eseguire sei operazioni al mese per mantenere una buona posizione nel mercato. Ecco perché i day trader hanno dovuto affrontare più difficoltà nel mantenere la loro posizione nel mercato poiché il loro livello di rischio è più alto degli swing trader e hanno dovuto impegnarsi di più nel mercato rispetto agli swing trader.

Fatica:

I day trader sono più stressati in quanto devono mantenersi costantemente impegnati con la situazione del mercato. Hanno bisogno di una grande conoscenza dei movimenti di mercato e hanno avuto un grande livello di pazienza. Un day trader deve essere più concentrato sul proprio lavoro. D'altra parte, gli swing trader non sopportano molta pressione e non possono dire di essere molto concentrati rispetto ai day trader.

2. COME FUNZIONA?

Che cos'è lo swing trading?

"Lo swing trading è una tecnica utilizzata per acquistare e vendere azioni".

Come funziona?

Swing trading è una delle strategie di trading più utilizzate e comuni utilizzate in quasi tutti i mercati, inclusi forex, trading di futures, azioni e molto altro. Lo swing trading riguarda l'acquisto e la vendita di azioni, l'acquisto dal mercato e la vendita per guadagni. A tal fine, i trader si affidano principalmente all'analisi tecnica per individuare buone opportunità di vendita e di acquisto.

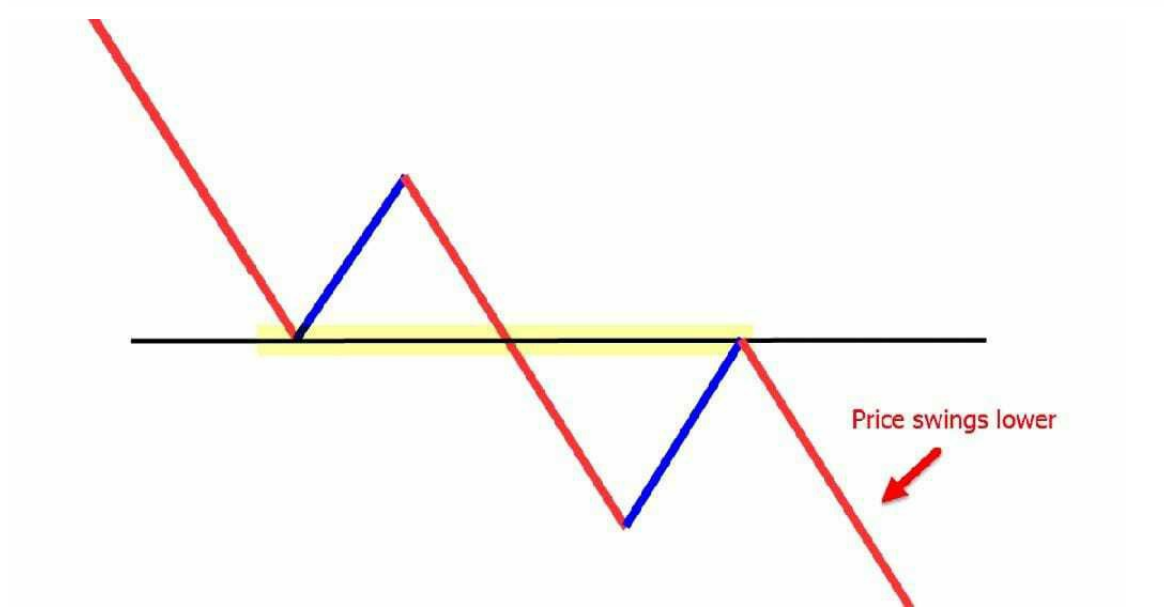
Successivamente, lo swing trading si basa principalmente su asset fondamentali poiché è un grande deterrente per movimenti di prezzo significativi. A volte ci vorranno giorni e persino settimane. Aiuta anche a migliorare l'analisi commerciale. Pertanto, un trader può verificare dove gli asset fondamentali sono favorevoli o meno, oppure potrebbe potenzialmente migliorare invece di fare affidamento sui modelli ribassisti. Gli swing trader utilizzano l'indicatore di analisi tecnica per identificare il prezzo dello swing trade e determinare se il prezzo di un'azione aumenterà nel mercato o scenderà. Sperimentando indicatori tecnici, gli swing trader non sono preoccupati per il valore a lungo termine del titolo.

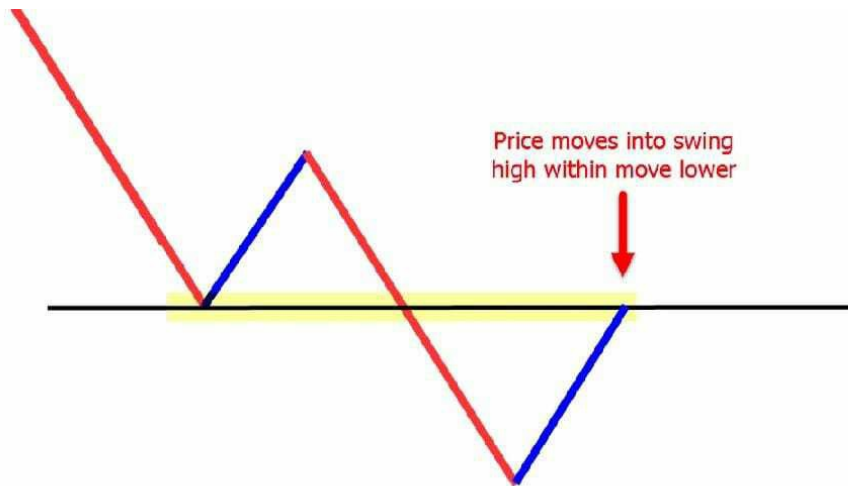
Come fare scambi swing con le tendenze?

Lo swing trading è uno dei migliori trading solidi e ha una delle tendenze ovvie nelle strategie di trading. Per i principianti, è necessario comprendere la sua importanza nel mercato che una volta appreso come investire i propri soldi, offrirà molte opportunità di trading elevate con un alto rialzo. Per un principiante, è obbligatorio avere una conoscenza sufficiente del mercato. Il primo passo che ogni principiante dovrebbe fare è identificare le esigenze del mercato, dovrebbe conoscere le tendenze del mercato.

Per ottenere una buona posizione nel mercato, ogni trader dovrebbe seguire le migliori tendenze, qualsiasi tendenza che vada in cima alla lista che se le mostri a un bambino, sceglierebbero chiaramente quella giusta i cui prezzi stanno ottenendo

superiore o inferiore.





Uno degli errori più comuni che ogni principiante fa quando cerca di oscillare il commercio all'interno di una tendenza non è entrare con il giusto punto di oscillazione.

Suggerimenti per i principianti

Quando dovresti scegliere lo swing trading:

Lo swing trading è più facile del day trading, ma per vedere se scegliere lo swing trading è una buona decisione per te o meno, dovresti raccogliere tutti i punti e vedere quale ti darà più profitto.

Assicurati di essere pronto per trascorrere giorni, settimane e mesi questo trading come swing trading contiene un periodo più lungo per il trading.

Se hai già un lavoro e vuoi investire i tuoi soldi per entrate extra, ma non hai abbastanza tempo per sederti davanti a un computer, quindi lo swing trading è l'opzione migliore per te.

Non è necessario fare un monitoraggio costante e tenere d'occhio il mercato attività.

Lo swing trading ti darà una vita meno stressante e ridurrà il rischio livello.

Come principiante, dovresti sempre avere un piano e attenerti ad esso poiché è il natura dell'acquisto e della vendita che ci saranno alti e bassi, ma dovresti attenerti al piano.

Prima di iniziare lo swing trading, dovresti esercitarti guardando mercati. Avere conoscenza del pre-trading ti aiuterà a guadagnare

la perdita.

Non aspettarti di avere soldi su una nota veloce; concentrati solo sul tuo obiettivo e lascia che il processo affondi.

Avere una buona osservazione ti aiuterà a investire denaro nel modo giusto stock, quindi se vuoi guadagnare soldi sul primo stock, allora dovresti sapere come usare gli indicatori di analisi tecnica.

È importante che un principiante tenga a mente mentre si scambia l'ingresso e strategia di uscita. Prima di entrare in un mercato, dovremmo controllare tutto e tenerci aggiornati con lo studio dei grafici, annotando il prezzo che probabilmente il titolo raggiungerà nel suo swing attuale.

La maggior parte dei trader (come i frequentatori di feste) trova più favorevole scegliere come uscire prima che siano entrati nel commercio.

Ci saranno più possibilità di perdere denaro nelle prime fasi di trading, ma un buon trader dovrebbe attenersi ad esso e dovrebbe acquisire competenze di base, perfezionarle, apprendere nuove tecniche tecniche e migliorare le proprie conoscenze. È importante non arrendersi in una condizione.

Dovrebbe leggere libri e seguire i consigli di swing trading per i principianti.

Piattaforme di trading

Per uno swing trading di successo, il trader dovrebbe selezionare le azioni giuste per realizzare un profitto sul mercato. Pertanto, i trader possono anche optare per azioni con una grande capitalizzazione di mercato. Gli asset più attivi saranno tra i principianti e saranno quelli più attivamente scambiati sui migliori exchange. Se parliamo delle piattaforme per lo swing trading, allora, da un lato, c'è un mercato ribassista e dall'altro c'è il mercato rialzista. Allo stesso modo, c'è un mercato stabile tra i due. Tuttavia, lo swing trading in un mercato ribassista o rialzista è piuttosto diverso da quello stabile.

Ad esempio, le azioni del mercato potrebbero tendere ad aumentare e diminuire utilizzando vari modelli in entrambi questi mercati, a differenza di un mercato stabile in cui un modello simile potrebbe essere mantenuto per settimane o mesi. Nel mercato ribassista e nel mercato rialzista, lo slancio può far sì che le azioni vengano scambiate in un modo specifico per un lungo periodo. In quella situazione, per ottenere il massimo profitto, un trader dovrebbe operare secondo il trend a lungo termine. E per sperimentare quel profitto, uno swing trader deve regolare correttamente il mercato attuale, sia esso ribassista o rialzista.

La media mobile semplice (SMA) consiglia fasi di supporto e resistenza specifiche. Mostra anche modelli rialzisti e ribassisti. Quando il titolo supera il livello di supporto, segnala un momento virtuoso per l'acquisto e, una volta raggiunto il livello di resistenza, è un buon momento per fare trading. I metodi rialzista e ribassista segnalano anche i punti di prezzo di entrata e di uscita. Pertanto, un'altra forma di SMA è la media mobile esponenziale (EMA) e si concentra su punti dati all'avanguardia. I segnali di tendenza, così come i punti di ingresso o uscita presentati dall'EMA, sono più rapidi rispetto alla SMA. I periodi 9, 13 e 50 EMA possono essere utilizzati per migliorare i punti di ingresso del tempo.

Ad esempio, quando il prezzo di un'azione incrocia questi EMA, mostra una tendenza rialzista. Pertanto, potrebbe esserci probabilmente un problema con la tendenza ribassista. Ad esempio, quando l'EMA 9 attraversa l'EMA 13, mentre l'EMA 13 è al di sopra dell'EMA a 50 periodi, segnala un'opportunità di acquisto.

Che cos'è il day trading?

"Vendere e acquistare un titolo più volte in un giorno."

Come funziona il day trading?

Sembra molto interessante fare trading, e chi volesse entrare a far parte di questo sistema di trading potrebbe vederlo con più interesse perché sembra molto facile e interessante, ma in realtà la vita di un trader è come vivere una vita al limite. Nel day trading, non abbiamo idea di cosa accadrà nel momento successivo; la scansione degli eventi imprevedibili si verifica in qualsiasi momento. L'amara realtà del trading è che la maggior parte dei giorni sono asciutti e abbastanza ordinari, e nel momento successivo vedrai il tuo nome in cima alla lista. Nel day trading, l'elevata velocità di esecuzione è molto importante in quanto potresti vedere l'elevato numero di operazioni che potresti fare in un giorno per abbinare te stesso al prezzo di mercato di cui hai bisogno per abbinare il livello del mercato. Per far funzionare il tuo trading, è molto importante abbassare i tassi di commissione. Sarà un day trading più redditizio in quanto avrà un tasso di commissione inferiore. I principianti dovrebbero mantenere la conformità normativa nel loro trading. I day trader di solito iniziano con zero posizioni nei loro portafogli tipici e i day trader fanno trading così frequentemente che alla fine della giornata hanno chiuso tutte le loro operazioni.

Alcuni trader giornalieri lavorano manualmente, facendo trading di ora in ora utilizzando un grafico. Altri impostano un processo automatico che effettua ordini di acquisto e commercio per loro. Mentre i day trader in realtà non guardano

dati fondamentali, sono interessati alla volatilità dei prezzi, alla liquidità, alle tendenze di rottura e al volume di scambi della giornata che cambierà significativamente il prezzo di un'azione.

PROMEMORIA:

Per il day trading, il forex è la piattaforma migliore. Per esercitarsi nel trading di forex e futures, un trader dovrebbe utilizzare la funzione Ninja Trader Replay. Aiuterà in praticando i giorni di trading storici come se stessi facendo trading sul serio.

Come fare trading giornaliero con le tendenze?

È molto necessario per un trader seguire le tendenze del mercato per ottenere una posizione forte in un mercato. Fare trading contro il trend e non seguirlo è uno dei motivi più comuni per affrontare il fallimento nel trading. La scarsa qualità della tendenza e il mancato rispetto di una tendenza sono una delle ragioni principali per non attirare l'attenzione dell'acquirente. Coloro che seguono la qualità e le tendenze forti hanno maggiori possibilità di ottenere successo nel trading. Nel trading, la tendenza aumenta la direzione generale del prezzo di un'azione. C'è un'altra possibilità di avere trader che non sono mentalmente e fisicamente attivi tutto il tempo per rispondere a un gran numero di azioni o variazioni di prezzo. Per coloro che non possono gestire le cose da soli, possono utilizzare uno stock screener; aiuterà il trader a restringere il numero di azioni e ridurre le dimensioni in modo da rendere le cose più facili nel lavorare per te. Se segui le tattiche di tendenza, scambia solo azioni che hanno una tendenza di tendenza. Se disponi di uno screener di azioni o di un assistente, possono aiutarti a isolare le azioni in quella tendenza in modo da avere un elenco di azioni e puoi facilmente applicare le tue strategie di trading giornaliero attraverso di essa. Qualunque azione tu scelga di negoziare rappresenta il tuo stile di trading, e un buon stile di trading ha sempre mantenuto una buona posizione nel mercato. e puoi facilmente applicare le tue strategie di day trading attraverso di esso. Qualunque azione tu scelga di negoziare rappresenta il tuo stile di trading, e un buon stile di trading ha sempre mantenuto una buona posizione nel mercato. e puoi facilmente applicare le tue strategie di day trading attraverso di esso. Qualunque azione tu scelga di negoziare rappresenta il tuo stile di trading, e un buon stile di trading ha sempre mantenuto una buona posizione nel mercato.

Per le azioni, il momento migliore per il day trading è la prima da una a due ore dopo l'apertura e l'ultima ora prima della chiusura del mercato. Dalle 9:30 alle 11:30 Le possibilità di una buona giornata di trading di solito iniziano un po' prima di altre nel mercato azionario. Il mercato forex viene scambiato 24 ore al giorno per tutta la settimana. L'EURUSD è la famosa coppia di day trading. Questa coppia di valute di solito registra volumi di trading migliori tra 1 AM e 12 PM EST. I mercati di Londra sono aperti in questi orari. I day trader scambierebbero in queste ore. Le ore dalle 7:00 alle 10:00 EST in genere generano le maggiori variazioni di prezzo perché entrambi i mercati di Londra e New York sono aperti durante questa durata. Pertanto, questo è molto comune

e tempo attivo per i day trader.

Suggerimenti per i principianti

Quando dovresti scegliere il day trading:

Per un principiante è molto importante fare un elenco di tutte quelle cose che possono aiutarlo a diventare un buon trader ed elencare anche tutti quei punti negativi che possono creare un problema. Scegliere il momento giusto per fare trading è molto importante quando si è mentalmente e fisicamente pronti a dedicarsi a questo percorso.

Innanzitutto, verifica se stai soddisfacendo tutti i requisiti della SEC e

Regola del trader del giorno del modello FINRA.

Concediti un minuto e pensa se sei pronto a sederti davanti di un computer da due a quattro ore ininterrotte per tenersi aggiornati sulla situazione attuale del mercato.

Serve molta pazienza per diventare un buon trader; ecco perché è molto importante per mantenere alto il livello di pazienza.

Un commerciante non ha bisogno di una laurea o di un diploma professionale per un giorno commercio. Né un trader ha bisogno di imparare migliaia di libri.

La disciplina è molto importante per il trading; se inizi a violare tutto, allora non puoi mai scambiare le tue azioni sul mercato.

Un day trader dovrebbe avere la capacità di correre un rischio e gestire le cose se affronta la perdita.

Non dovrebbe stressarsi per le cose ma dovrebbe controllarle in modo positivo maniera.

Dovresti avere un computer per fare trading, avere due monitor è preferibile ma non necessario. Il tuo computer dovrebbe avere una grande memoria e un processore veloce in quanto può darti uno svantaggio se il processore o il software non sono buoni.

Avere una buona connessione a Internet è un'altra cosa molto importante se tu perdere la connessione durante uno scambio di quanto perderai anche lo scambio.

Per iniziare la tua carriera di trading, è necessario selezionare la piattaforma giusta. Poiché i principianti non hanno una conoscenza sufficiente della piattaforma giusta, non sei a conoscenza di uno stile di trading ben sviluppato.

Nel day trading, un trader non ha bisogno di fare trading tutto il giorno. forse lo farai scopri più stabilità facendo trading solo due o tre ore al giorno.

Prendi nota del tuo grafico di profitti/perdite in pips (forex), punti (future) o centesimi (azioni) su base giornaliera perché si tratta di cifre scalabili. Annotare le cifre in dollari può farti confondere perché ci saranno più possibilità che il saldo del tuo conto possa fluttuare nel tempo, risultando in operazioni più grandi o più piccole.

Salva tutti i dati raccolti con un nome, mese-giorno-anno.

Crea una cartella sul tuo computer e archivia i file salvati lì per recensioni successive.

Ogni fine settimana, esamina i dati raccolti dalla settimana precedente.

Nota dove hai commesso un errore e cosa devi migliorare in te stesso.

I day trader dovrebbero praticare tutti i problemi precisi in un conto demo durante le ore non commerciali.

NOTA:

Per avere una buona posizione nel mercato, il trader dovrebbe trovare la ripetizione modelli che stanno facendo continuamente profitto nel mercato.

Piattaforme di trading

Un principiante ha sempre in mente la piattaforma di trading ogni volta che pensa di iniziare a fare trading dove dovrebbe operare o meno, quale piattaforma sarà più vantaggiosa per lui. Il futuro del commercio si basa più spesso sugli indici e sulle materie prime. Un principiante potrebbe scambiare oro, petrolio greggio o movimenti S&P. Non tutti i mercati sono buoni; cambia e si riduce a ciò che scambi e ciò che puoi permetterti al momento. Esistono molti mercati per il trading che possono aiutare i principianti a raggiungere i propri obiettivi, ma trovare quello giusto su cui investire i propri soldi è molto importante. I day trader sono ammirevoli amanti del rischio. Prendono rischi in quella zona in cui non possono permettersi di correre rischi. Tuttavia, utilizzano questa piattaforma di trading. I day trader devono disporre di una piattaforma veloce e affidabile, ricca di strumenti e funzionalità per garantire un'esperienza di trading ottimale.

Ci sono migliaia di azioni sul mercato tra cui scegliere, come decidi quale ti darà più profitto o su cui ti concentrerai per il day trading? Può essere fonte di confusione per un principiante che cerca di capire quello giusto. Un giorno i trader trovano facilmente nuovi titoli da scambiare ogni giorno o vanno a caccia di titoli che stanno uscendo dagli schemi. Pertanto, altri cercano titoli che

rottura dei livelli di supporto o resistenza o sono i più volatili. Un principiante dovrebbe tenere a mente queste cose mentre sceglie un mercato, che quando hai raccolto un mercato per il tuo investimento, dovresti avere l'attrezzatura adeguata e la configurazione del software e la conoscenza delle merci per il day trading. Quando inizi a pensare al trading, devi sapere come controllare il rischio. I day trader possono controllare il rischio in due modi: rischio commerciale e rischio giornaliero. Il rischio commerciale è fino a che punto sei disposto a correre il rischio su ogni operazione. Idealmente, rischi circa l'1% o meno dei tuoi soldi su ogni operazione.

Un principiante può anche iniziare il suo commercio con una piccola somma di \$50, anche se se può investire di più è consigliabile iniziare con di più. Mentre alcuni mercati di trading richiedevano \$ 1.000 per iniziare. Un day trader richiede almeno \$ 25.000 per scambiare le sue azioni. La necessità di avere più capitali per il commercio giornaliero di azioni non lo renderà migliore o peggiore sul mercato rispetto agli altri. È essenziale per il trading mantenere buona la tua posizione nel mercato, perché devi mantenere il tuo conto fino a \$ 25.000, ma se stai affrontando continui fallimenti, allora questo mercato non è un buon posto per te.

3. SWING TRADING E GIORNO TRADING: TATTICHE E STRATEGIE

Che cos'è la strategia?

"La strategia è un'azione o un piano che viene utilizzato per raggiungere uno o più obiettivi organizzativi."

UNA strategia di trading è una procedura attraverso la quale un trader vende e acquista il titolo e si basa su regole predefinite utilizzate per prendere decisioni di trading. Qualsiasi tipo di processo di negoziazione nel mercato di solito include un piano di trading e investimento ben ponderato che specifica la tolleranza al rischio, l'implicazione fiscale e la capitalizzazione degli oggetti. Applicare o pianificare una strategia nel trading significa che un trader dovrebbe cercare e adottare le migliori pratiche e idee e poi le segue.

PROMEMORIA:

La strategia prevede tre fasi:

Pianificare uno scambio, piazzare uno scambio e poi eseguirlo.

Un trader dovrebbe capire il livello di rischio che può correre e poi decidere cosa è appropriato fare per lui. Le strategie di trading si basano principalmente su fondamentali o metodologico. Per evitare pregiudizi di finanza comportamentale e per assicurarsi risultati coerenti, vengono impiegate strategie di trading. Anche se è molto difficile sviluppare una strategia di trading perché ci sono più possibilità di avere il rischio di diventare eccessivamente dipendenti da una strategia.

Di quante strategie ha bisogno un trader?

Un trader dovrebbe utilizzare solo una o due strategie per un trading di successo. È un modello di acquisto e vendita delle azioni che ogni trader utilizza nella sua routine quotidiana, che delinea quando un trader entrerà e uscirà dal mercato. La strategia di trading consente al trader di vedere le opportunità di trading in modo obiettivo. Consente inoltre al trader di vedere come sono andate le operazioni e i trader in passato.

Tipi di strategie:

Esistono quattro tipi di stili di trading.

1. Scalping.
2. Negoziazione giornaliera.
3. Negoziazione di posizione.
4. Swing trading.

Stili di trading, time frame e loro tempo di una cornice di periodo è data sotto:

Stile di trading	Lasso di tempo	Periodo di Portafoto
scalping	A breve termine	Secondi o minuti
Giorno di negoziazione	A breve termine	Per un giorno (massimo)
Trading di posizione	Lungo termine	Settimane, mesi o anni.
Swing Trading	Medio termine	Giorni o settimane

Questi stili di trading sono i quattro principali tipi di trading maggiormente utilizzati nel mercato forex.

Strategia di trading giornaliero

Le strategie di day trading sono importanti per il trader quando cerca di capitalizzare su piccoli movimenti di prezzo. Una strategia di trading aiuta il trader a capire come tra migliaia di azioni, un lettore può decidere o scegliere quella giusta. Questo libro aiuterà i principianti a comprendere la situazione del mercato e aiuta a concentrarsi su quale strategia aiuterà il trader. A volte i principianti si confondono a causa della loro esperienza zero all'inizio e perdono le loro speranze, ma qui cercheremo di porre fine alla confusione dei principianti del trading prima che inizi effettivamente nelle loro menti. Una strategia coerente ed efficace nel day trading si basa sull'utilizzo di grafici, indicatori tecnici, analisi tecniche approfondite e schemi; aiuta a prevedere i futuri movimenti dei prezzi nel trading. Questo libro ti fornirà una ripartizione dettagliata delle strategie di trading per principianti.

È essenziale per un principiante conoscere il concetto di base del trading perché potrebbe impantanare un trader in un mondo complesso di indicatori altamente tecnici, ecco perché concentrazione e conoscenza sono entrambi importanti per un semplice day trading

strategia. Avere pazienza e controllo è molto importante per il day trader perché ci saranno giorni in cui si rivelerà che saranno giorni variabili nel trading. Uno degli errori più comuni che ogni principiante di solito fa è correre il rischio di fare trading troppo presto sapendo che può danneggiarli finanziariamente, cercando comunque di ottenere un prezzo e una posizione migliori nel mercato e presumendo che il commercio li attiverà in futuro . Questo è l'errore più grande che ogni principiante di solito ha fatto.

Fondamenti strategici di base che ogni giorno il trader dovrebbe usare:

Un commerciante non dovrebbe aspettarsi di fare una fortuna se sta solo allocando un un'ora o due al giorno per fare trading. Un trader deve monitorare continuamente i mercati e continuare a cercare opportunità di trading.

Prima di iniziare a fare trading, devi decidere quanto stai pronto a rischiare. Un trader dovrebbe tenere a mente che i trader di maggior successo non metteranno più del 2% del loro investimento sulla linea per operazione.

Un trader dovrebbe prepararsi per alcune perdite se vuole essere in giro quando il vincitore inizia a rotolare.

Un trader dovrebbe assicurarsi di rimanere aggiornato sugli eventi e notizie di mercato che influenzeranno il tuo asset, come un cambiamento nella politica economica. Un trader può scoprire una vasta gamma di risorse economiche e commerciali online che lo terranno al corrente.

Per il trading, solo avere la conoscenza delle complessità del mercato non lo è abbastanza; un commerciante dovrebbe essere informato su tutto.

Il mercato commerciale diventerà volatile quando si apre ogni giorno e durante commercianti giornalieri praticati che sono in grado di leggere i contorni e guadagnare, un trader dovrebbe aspettare il suo tempo

È più difficile di quanto sembri tenere a bada le tue emozioni quando sei a cinque caffè e stai guardando lo schermo da ore. Un trader dovrebbe lasciare che la matematica, la logica e la strategia di trading lo guidino piuttosto che i nervi, la paura o l'avidità.

Un trader deve avere gli strumenti tecnici all'inizio, ma anche i migliori luogo per sperimentare nuove strategie per i trader avanzati è il conto demo. Diversi account demo non sono soggetti a restrizioni, quindi non sono previsti limiti di tempo.

Componenti importanti per il day trading:

Che si tratti di strategie di day trading automatizzate o di tattiche avanzate e per principianti, un trader dovrà prendere tre componenti essenziali; liquidità, volatilità e volume. Se un principiante vuole creare denaro facendo piccoli movimenti di prezzo, scegliere il titolo giusto è vigoroso. Questi tre elementi aiuteranno il trader a prendere una decisione.

1. Liquidità:

Ciò consente al trader di entrare e uscire improvvisamente dalle negoziazioni a un prezzo accattivante e stabile. Strategie su materie prime liquide, ad esempio petrolio greggio, focus su oro e gas naturale, ecc.

2. Volatilità:

Attraverso questo, ogni trader principiante imparerà a conoscere la loro potenziale gamma di profitto. Maggiore è la volatilità, maggiore è il profitto o la perdita che un trader può realizzare. Il mercato delle criptovalute è un esempio di elevata volatilità.

3. Volume :

Questa misurazione aiuterà il principiante a sapere quante volte il titolo/asset è stato scambiato entro un determinato periodo di tempo. Per i principianti di un day trader, questo è noto come "volume di trading giornaliero medio". Un volume elevato aiuta a sapere che c'è un interesse significativo nell'asset o nella sicurezza. La crescita del volume è spesso un indicatore che il prezzo sale o scende, si avvicina rapidamente.

Quanto rischio è coinvolto nel day trading?

Secondo gli esperti di (OTA) Online Trading Academy, è una realtà che le situazioni di day trading gestite in un solo giorno lo stanno rendendo davvero più sicuro piuttosto che più rischioso.

"Uno dei modi migliori per controllare il rischio è limitare la durata dell'operazione. Più a lungo sei in una posizione, maggiore è la probabilità che il prezzo possa muoversi contro di te. Di giorno, elimini il trading durante la notte e il rischio del fine settimana, soprattutto quando fai trading su mercati che si chiudono, come le azioni".

– Brandon Wendell, CMT

Poiché è un dato di fatto che i day trader non mantengono le loro posizioni durante la notte, di solito evitano la probabilità di una sorpresa in un mercato estero, notizie finanziarie sfavorevoli o un rapporto sui redditi che esce una volta che i mercati sono chiusi. Anche se per numerosi titoli viene offerto il trading fuori orario, il mercato è alto ed è possibile che la posizione lo facciaspacco

fuori uso e aprire a un prezzo notevolmente inferiore il giorno successivo dopo un'esperienza notturna negativa.

Inoltre, il day trading tende a facilitare, non ad aumentare la volatilità del mercato. I day trader di solito cercano i loro guadagni in piccoli movimenti di prezzo al rialzo o al ribasso. Le loro operazioni quotidiane offrono liquidità, che aiuta il marketing a correre facilmente, rispetto ai mercati negoziati casualmente che sono soggetti a drammatiche oscillazioni dei prezzi. Il day trading non è un modo per diventare ricchi all'istante. Il day trading è un approccio di investimento tradizionale utilizzato da molte organizzazioni e dalle istituzioni ben istruite che lo hanno adottato come professione.

Negli anni '90, il day trading non aveva una buona reputazione e, a quel tempo, molti principianti iniziarono il day trading. Hanno iniziato a saltare su diverse piattaforme, comprese le piattaforme di trading online, senza applicare le strategie azionarie. Credevano di poter gestire il mercato senza avere la conoscenza del mercato e fare una fortuna nelle compravendite di azioni con il loro piccolo sforzo. Questo ha dimostrato che si sbagliavano.

Di cosa hai bisogno per iniziare il day trading?

Per un principiante del day trader, è importante disporre di attrezzature tecniche a casa tua. La maggior parte dei principianti pensa che il trading quotidiano richieda attrezzature pesanti e costose e alti investimenti di capitale; questa non è la realtà.

Ecco un elenco di oggetti di cui solitamente ogni trader ha bisogno per il trading.

Tecnologia:

Per il trading, i trader non vogliono un computer sovralimentato con una dozzina di monitor da scambiare. Hanno solo bisogno di un laptop o di un computer.

Connessione internet:

È molto importante disporre di una buona connessione a Internet; aiuta il commerciante a elaborare il suo ordine in modo tempestivo. La maggior parte dei fornitori di cavi e persino di satelliti offre una larghezza di banda sufficiente per connettersi alle centrali. Per questo sono sufficienti pacchetti tipici di 20 Mbps di Internet via cavo. La maggior parte dei trader utilizza persino le proprie connessioni di telefonia mobile da 5 Mbps a 20 Mbps, ma ciò non è suggerito. La connessione lenta del telefono cellulare può causare ritardi nelle transazioni e può causare perdite impreviste.

Piattaforma di trading:

I trader dovrebbero stare attenti nel mercato del trading perché ci sono molti broker online seduti online per ingannare i principianti. Offrono ai principianti

i loro servizi ma indirizzano gli ordini sui market maker che costano denaro aggiuntivo, spesso rinviando l'elaborazione. È facile eseguire analisi commerciali ed effettuare ordini molto rapidamente e correttamente. La versione basata sul web è meno affidabile della versione scaricata, la versione scaricata offre più funzionalità.

Abilità

Molte persone sono sostenitori della ricerca di un'istruzione. Il problema principale è quello; l'istruzione da sola non è sufficiente se non viene utilizzata correttamente nel mercato. Sebbene sia essenziale avere informazioni sui mercati, su come funzionerà, su come leggere il prezzo e su come offrirà un vantaggio, è necessaria anche abilità per ottenere risultati stabili.

Costruire un'abilità richiede pratica ed esperienza, ma cercare di acquisire abilità di trading senza supervisione può essere un processo lungo e spesso fastidioso. Per molti trader, lavorare e imparare dall'esperienza di un consulente è il modo migliore per migliorare qualsiasi abilità e apprendere strategie di trading e investimento che riducano il rischio. Anche i famosi commercianti come Paul Tudor e Warren Buffett Jones avevano bisogno di mentori. Il signor Buffett ha lavorato sotto Benjamin Graham e il signor Jones ha lavorato sotto Eli Tullis.

Strategie di trading giornaliero

Strategie

1. Evasione

Nel trading le strategie di breakout si concentrano dopo che il prezzo ha superato un livello specificato sul grafico, con un volume migliorato. Il trader di breakout entra in una situazione lunga una volta che la sicurezza o l'asset supera la resistenza. In caso contrario, un trader entra in un sito short dopo che il titolo è sceso al di sotto del supporto.

Una volta che un titolo o un'attività viene scambiata oltre la barriera dell'importo quantificato, la volatilità aumenta frequentemente e l'importo tenderà spesso alla via del breakout.

Un trader deve prendere lo strumento giusto per fare trading. Mentre lo fai, tieni presente il livello di resistenza e il supporto dell'asset. Più spesso il prezzo ha colpito questi fatti, più autentici e vitali diventano.

Pianifica i punti di entrata e di uscita:

Questa parte del trading è buona e diretta. Prezzi impostati nelle vicinanze e superiori

i livelli di resistenza vogliono un posto ribassista. I prezzi impostati vicino e al di sotto di un livello di manutenzione vogliono un posto rialzista.

Utilizzo delle ultime prestazioni dell'asset per stabilire un obiettivo di prezzo ragionevole. L'utilizzo di modelli grafici nel trading renderà questo processo ancora più preciso. Un trader può analizzare le oscillazioni dei prezzi medi correnti per generare un obiettivo. Se le oscillazioni medie sono state di 3 punti rispetto alle ultime oscillazioni di prezzo, questo sarebbe un obiettivo praticabile. Una volta raggiunto questo obiettivo, puoi uscire dal trade e goderti il profitto.



2. Scalping

Lo scalping è una delle migliori strategie, utilizzata principalmente dai trader. È per lo più utilizzato ed è popolare nel mercato forex. Cerca di trarre vantaggio dalle variazioni minime dei prezzi e la sua forza trainante è la quantità. Cercherai di vendere lo stock non appena lo stock diventa redditizio. Questa è una tecnica entusiasmante e veloce per fare trading, ma poi di nuovo, può essere rischiosa. Devi avere un'alta probabilità di trading per uniformare il rapporto di ricompensa rispetto al rischio basso. Un trader dovrebbe essere alla ricerca di strumenti volatili, liquidità interessante ed essere puntuale. Non puoi aspettare il mercato; devi chiudere le operazioni in perdita il prima possibile.



3. Slancio:

Questa strategia è popolare tra tutte le strategie di trading per principianti; questa strategia ruota attorno all'azione sul riconoscimento di grandi movimenti di tendenza con il supporto di fonti di notizie e volumi elevati. Per l'ampia opportunità, c'è sempre almeno uno stock che si muove intorno al 20-30% ogni giorno. Un trader semplicemente mantiene la posizione finché non vede segni di inversione e poi esce. Altrimenti, può far sparire il calo dei prezzi. In questo modo aggira il suo obiettivo di prezzo non appena il volume inizia a ridursi.

Questa è la strategia più semplice ed efficace se usata correttamente. Tuttavia, un trader deve assicurarsi di essere a conoscenza delle notizie imminenti e degli annunci di reddito. Solo un piccolo numero di secondi su ogni operazione farà la differenza per i tuoi profitti di fine giornata.



4. Inversione

Sebbene questa strategia sia oggetto di accesi dibattiti e potenzialmente pericolosa quando si tratta di essere utilizzata dai principianti. Il reverse trading è utilizzato in tutto il mondo. È anche noto come trend di pullback, trading di tendenza e strategia di mean reversion. Questa strategia affronta la logica di base in quanto il trader mira a negoziare contro il trend. Un trader deve essere in grado di classificare correttamente i possibili pullback, oltre a calcolare la loro forza. Per farlo in modo efficace, un trader deve aver bisogno di un'esperienza e una conoscenza del mercato approfondite.

La strategia del "pivot giornaliero" viene misurata come un caso unico di reverse trading, poiché si concentra sulla vendita e sull'acquisto dei pullback/reverse massimi e minimi giornalieri.



5. Utilizzo dei punti pivot

Una strategia di punto pivot di trading giornaliero può essere strana o fantastica nel trading per agire su livelli critici di supporto e/o resistenza che la identificano. È per lo più utile nel mercato forex. Inoltre, i punti pivot possono essere utilizzati dai trader rangebound per riconoscere i punti di ingresso, mentre i trader di trend e breakout possono utilizzare i punti pivot per individuare i livelli chiave che devono interrompere una mossa per essere considerata un breakout.

Calcolo dei punti pivot

Un punto pivot è ben definito come un punto di rotazione nel day trading. Un trader principiante può utilizzare i prezzi del minimo o del massimo del giorno precedente e, più il prezzo di chiusura di un titolo per analizzare il punto di rotazione.

Tieni presente che se analizzi un punto pivot utilizzando le statistiche sui prezzi da un intervallo di tempo piuttosto breve, la precisione viene spesso ridotta.

Quindi, in che modo un day trader analizzerà/calcolerà il punto pivot?

- Punto di articolazione centrale (P) = (Alto + Basso + Chiudi) / 3

Ora i day trader possono analizzare i livelli di resistenza e supporto utilizzando il punto pivot. Per fare ciò un trader deve utilizzare le seguenti formule:

- Prima Resistenza (R1) = (2*P) - Basso
- Primo Supporto (S1) = (2*P) - Alto

Il secondo livello di resistenza e supporto viene quindi calcolato come segue:

- Seconda Resistenza (R2) = P + (R1-S1)
- Secondo Supporto (S2) = P - (R1-S1)

Applicazione

Quando applicato praticamente nel mercato FX, ad esempio, un principiante troverà il range di trading per la sessione che avrà luogo frequentemente tra il punto pivot e i livelli di resistenza e il primo supporto. La ragione di questo è avere un numero elevato di trader che giocano in questa gamma. Vale anche la pena notare perché questo è uno dei sistemi e degli approcci che possono essere applicati anche agli indici.

Ad esempio, può aiutare un principiante del day trader a formare un'efficace strategia di day trading S&P.

Limita le tue perdite

Questa è la cosa più importante da tenere a mente se stai usando un

marginale sta limitando la tua perdita. I requisiti sono spesso elevati per i day trader. Quando un day trader negozia con un margine, sarà sempre più suscettibile a bruschi movimenti di prezzo. Ciò significa il potenziale per un profitto maggiore, ma significa anche la probabilità di perdite sostanziali. Fortunatamente, un trader può utilizzare gli stop loss. Lo stop-loss controlla il rischio commerciale per il trader. In una piccola situazione, un trader può piazzare uno stop-loss al di sopra di un recente massimo; per buone situazioni grandi, puoi posizionarlo al di sotto di un minimo recente. Un trader può anche renderlo dipendente dalla volatilità.

Ad esempio, se l'importo di un'azione si muove di 0,05€ al minuto, puoi piazzare uno stop-loss di 0,15€ dopo il tuo ordine di entrata, lasciandolo oscillare (si spera nella direzione prevista).

Una strategia popolare nel day trading consiste nell'impostare due stop-loss. In primo luogo, un trader effettua un ordine fisico di stop-loss a un preciso livello di prezzo. Questo sarà il capitale massimo che puoi permetterti di perdere. In secondo luogo, puoi creare uno stop-loss mentale. Posiziona questo, nel punto in cui i tuoi criteri di ingresso, sono violati. Quindi, se lo scambio fa una svolta inaspettata, farai una rapida uscita.

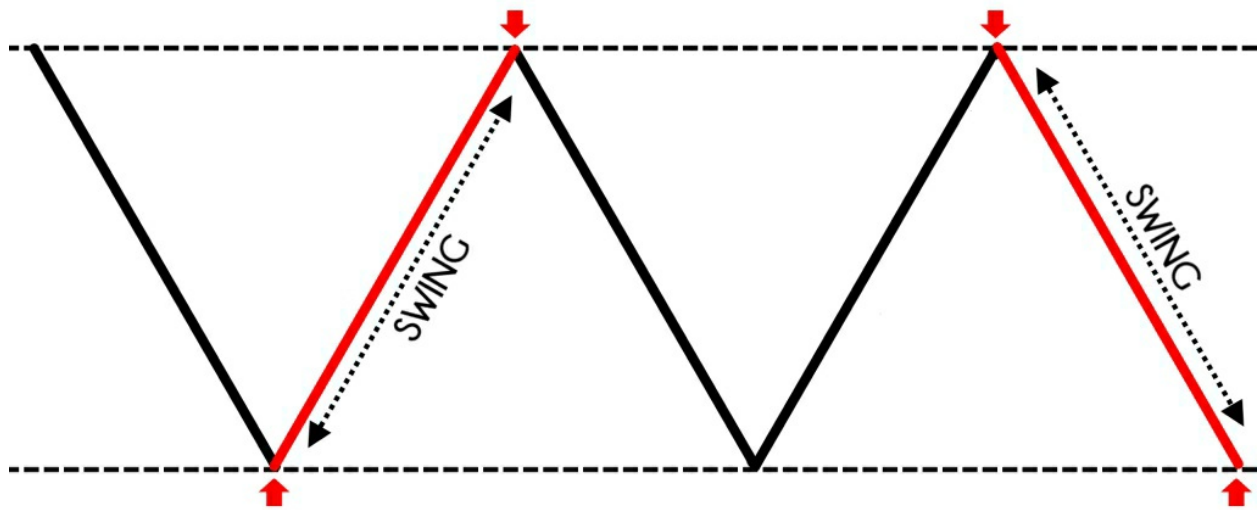
Strategie di trading Forex

Le strategie Forex sono rischiose per natura in quanto un trader deve accumulare i suoi profitti in un breve lasso di tempo. Un trader può applicare una qualsiasi delle strategie nel mercato forex.

Strategia di trading swing

Cos'è un trader swing?

Gli swing trader sono fondamentalmente quei trader che fanno trading per un paio di giorni o per settimane. Di solito funzionano per quattro ore (H4) e grafici giornalieri (D1) e possono utilizzare una combinazione di [analisi fondamentale](#) e [analisi tecnica](#) per monitorare il trading le loro decisioni. Che si tratti di una tendenza a lungo termine o che il mercato sia principalmente range-bound, non importa. Uno swing trader Forex non manterrà una posizione sufficiente per contare considerevolmente.



Nel mercato del trading, lo swing trader è il migliore da usare quando i mercati non vanno da nessuna parte una volta indici aumentare per più giorni, quindi decadere per i giorni successivi, solo per ripetere ancora e ancora gli stessi schemi complessivi. Lo swing trader ha diverse possibilità di cogliere i movimenti a breve termine al rialzo e al ribasso. Il problema con lo swing trading e il trend trading a lungo termine è che il successo si stabilisce riconoscendo correttamente quale tipo di mercato viene attualmente praticato. Il trend trading potrebbe avere la strategia perfetta per il mercato rialzista degli anni '90, mentre lo swing trading forse sarebbe stato il migliore per il 2000 e 2001.

Le medie mobili semplici (SMA) offrono livelli di resistenza e supporto, nonché modelli ribassisti e rialzisti. I livelli di supporto e resistenza possono aiutare il trader ad acquistare un titolo. I modelli di limite rialzista e ribassista segnalano idee sui prezzi in cui dovresti entrare e uscire dalle azioni. La media mobile esponenziale (EMA) è simile alla SMA che pone ulteriore enfasi sugli ultimi dati. L'EMA fornisce ai trader chiare indicazioni di tendenza, punti di entrata e di uscita più velocemente di una semplice media mobile (SMA). Gli swing trade possono utilizzare l'EMA per i punti di entrata e di uscita.

Uno swing trader tende a cercare i contorni dei grafici di più giorni. Alcuni degli schemi più comuni includono crossover a media mobile, modelli di testa e spalle, bandiere, triangoli e motivi a tazza e manico. Alla fine, ogni swing trader formula un piano e una strategia che gli danno un vantaggio su molte operazioni. Ciò include anche la ricerca di accordi commerciali che tendano a portare a movimenti prevedibili nel prezzo dell'attività. Non è così facile e

nessuna strategia o nessun accordo funzionerà ogni volta. Attraverso un rischio o una ricompensa favorevole, non è necessario vincere ogni volta. Più è promettente il rischio o la ricompensa di una strategia di trading, meno volte si desidera vincere per produrre un profitto completo su molte operazioni.

Una volta che si tratta di prendere profitti, gli swing trader, che siano principianti o professionisti, vorranno uscire il prima possibile dal commercio sulla linea del canale superiore o inferiore senza essere troppo definiti, il che può causare il rischio di perdere il migliore opportunità.

Nel libro del Dr. Alexander Elder, "Come Into My Trading Room: A Complete Guide to Trading" (2002), usa la sua comprensione del comportamento di un titolo al di sopra e al di sotto della linea di base per definire la strategia dello swing trader di "acquistare regolarità e vendendo mania" o "cortocircuitare la regolarità e coprire la depressione". Una volta che lo swing trader ha utilizzato l'EMA per riconoscere la tipica linea di base sul grafico azionario, il trader va lungo sulla linea di base una volta che l'azione è in salita e corta sulla linea di base una volta che l'azione è in discesa.

Pertanto, gli swing trader non stanno cercando di distruggere il fuoricampo con un singolo trade; non sono allarmati dal momento giusto per acquistare un titolo esattamente al suo fondo e vendere esattamente al suo massimo. In un ambiente di trading perfetto, i principianti dovrebbero aspettare che il titolo raggiunga la sua linea di base e confermi la sua strada prima di fare le loro mosse. La storia diventa più complessa quando è in gioco un trend rialzista o ribassista più duro. Il trader può inaspettatamente andare long una volta che il titolo scende al di sotto della sua EMA e fermarsi per far risalire il titolo in un trend rialzista, oppure un trader può [corto di azioni](#) che è rimasto al [di sopra dell'EMA](#) e aspetta che scenda se la tendenza che richiede tempo è al ribasso.

NOTA:

Molti trader credono di non poter fare scambi con il loro conto

è troppo piccolo.

Se un trader sta calcolando la sua posizione prima di ogni operazione e rischia un importo simile in ogni operazione, allora un trader dovrebbe essere in grado di svolgere un'operazione sia

lo stop è di 60 pips o 10.

Strategie di swing trading popolari:

Lo swing trading è spesso fatto all'interno delle tendenze, e questo è un modo comune; può anche essere effettuato con successo in mercati diversi. Uno swing trader analizza il profitto di prezzo e l'azione dalla maggior parte della prossima oscillazione del mercato. I mercati trascorrono molto più del loro tempo a raggiungere; poi lo fanno

creare tendenze chiare superiori o inferiori ed essere in grado di negoziare con successo mercati che vanno, è fondamentale.

Strategie di trading swing nel Forex:

Lo swing trading non è una strategia; è uno stile. Il lasso di tempo dello swing trading descrive questo stile e, al suo interno, ci sono strategie illimitate che possiamo usare per lo swing trading. Lo swing trading è uno stile che funziona su intervalli di tempo da brevi a moderati. Lo stile di swing trading si colloca tra i tempi molto lunghi del trading di posizione e i tempi brevi del day trading. Non è così breve da richiedere tutto il tuo tempo per osservare il mercato, ma è abbastanza breve da offrire molte possibilità di trading. Queste strategie non si limitano allo swing trading; è il caso della maggior parte delle strategie tecniche, supporto e resistenza sono le idee alla base.

Questi concetti possono dare a un trader due scelte all'interno della strategia di swing trading con, seguire il trend, o fare trading in controtendenza. Le strategie di controtendenza mirano a trarre profitto una volta che i livelli di supporto e resistenza reggono. Le strategie che seguono il trend cercano le possibilità che i livelli di supporto e resistenza si rompano.

Per il trading, gli swing trader possono utilizzare le seguenti strategie per opportunità di trading attuabili:

Strategia di trading swing 1: trading di tendenza

Quando si classifica un trend, è essenziale riconoscere che i mercati non tendono a muoversi in linea retta. Anche se alla fine sono di tendenza, si muovono su e giù a passo come mosse. Un trader dovrebbe riconoscere una tendenza al rialzo dal mercato, facendo massimi e minimi più alti, e una tendenza al ribasso riconoscendo minimi e massimi più bassi. Molte strategie di swing trading cercano di catturare e seguire una breve tendenza.

Strategia di trading swing 2: trading in controtendenza

La prossima strategia di swing trading è il trading in controtendenza e quindi fa il contrario della prima. Pratichiamo i principi simili in termini di tentativo di individuare tendenze relativamente a breve termine dall'edilizia, ma ora cerchiamo di trarre profitto dalla frequenza con cui queste tendenze tendono a crollare.

NOTA:

Trend rialzista = minimi e massimi decrescenti

Trend ribassista = minimi e massimi decrescenti

Un trader in controtendenza si sforzerebbe di cogliere l'oscillazione in questo periodo di inversione. Pertanto, il trader proverebbe a identificare la rottura del trend. In un trend rialzista, questo potrebbe essere un nuovo massimo seguito da una serie di fallimenti nel rompere nuovi massimi. Quindi andiamo a corto in attesa di un tale ritorno. È vero il contrario in una tendenza al ribasso.

È molto importante mantenere una forte disciplina incontrando il trend se il prezzo viaggia contro di te e se il mercato riprende il suo trend contro di te, allora dovresti essere pronto ad ammettere che ti sbagli e hai tracciato una linea sotto il trade.

Strategia di swing trading 3: una strategia di swing trading versatile

Se un trader desidera fare un tuffo ancora più profondo nello swing trading, oltre ad apprendere una strategia multiuso che anche i principianti possono utilizzare. Essere un trader versatile significa che un trader è in grado di negoziare qualsiasi strumento, in qualsiasi lasso di tempo o in qualsiasi direzione.

Migliorare le strategie di swing trading per i principianti:

Che swing possono fare i trader per aumentare le loro strategie?

Ci sono numerose cose che un trader può provare. La prima cosa è cercare di far corrispondere il commercio con la tendenza a lungo termine. Mentre il grafico orario aiuta lo swing trader a guardare anche un grafico a lungo termine per avere un'idea della tendenza di lunga data. Prova a fare trading solo una volta che la tua direzione corrisponde a quella che vedi come tendenza a lungo termine. Un altro modo per recuperare la tua strategia di swing trading è utilizzare un indicatore tecnico per confermare il tuo pensiero.

Una media mobile (MA) è un altro indicatore che puoi utilizzare per aiutarti. Un MA appiattisce i prezzi per dare una visione più chiara del trend. E poiché un MA include dati sui prezzi precedenti, è un modo semplice per confrontare il modo in cui i prezzi recenti si confrontano con i prezzi precedenti.

Gestire il rischio nello swing trading Forex:

Hai mai letto del trading sul Forex e del fatto che molti trader hanno perso soldi a causa di questo? Hai mai capito che questo vale anche per i trader di successo?

La realtà è che nessun trader vince il 100% delle volte. La maggior parte delle volte è

Succede che i principianti dello swing trading giudichino male il mercato, a volte si muova in modo imprevisto, a volte il trader potrebbe semplicemente sbagliare. Questo è il punto in cui la gestione del rischio e la gestione del denaro sono così importanti.

Nel trading, ma soprattutto nel Forex, un trader dovrebbe sapere come perdere prima di sapere come avere successo. E quando i trader parlano di avere abbastanza conoscenze su come perdere, dovrebbero anche sapere come perdere piccoli per vincere alla grande. Fondamentalmente, se un trader può raggiungere un rischio di swing trading, allora può chiudere in anticipo le operazioni in perdita, il che aiuterà a garantire che goda di profitti extra rispetto alle perdite.

Alcuni suggerimenti per gestire il rischio nello swing trading includono:

1. Perdita massima accettabile:

Nonostante il fatto che un trade voglia chiaramente che il suo prossimo trade realizzi un profitto, è importante pensare all'importo supremo che sei disposto a perdere su un trade. Nel momento in cui un trader sa che questo importo può impostare uno stop loss per chiudere spontaneamente il suo commercio se viaggia troppo lontano nella direzione sbagliata, questo lo aiuterà a proteggerlo quando non può essere automaticamente al tuo computer osservando ogni operazione.

2. Assumersi un rischio su qualsiasi operazione:

Indipendentemente dalle dimensioni del tuo conto di trading, devi evitare di rischiare l'intero saldo in un'operazione. Se un trader non lo segue, allora può perdere tutto. La regola generale è di non rischiare più del 2% del saldo del tuo conto su qualsiasi operazione.

3. Aumentare il saldo del conto per diversificare il rischio:

Nonostante il fatto che un trader possa essere in grado di aprire un conto a partire da 300€, è meglio iniziare a utilizzare una somma maggiore. Ciò significa che un trader avrà una quantità adeguata del tuo conto per scambiare una varietà di attività ed espandere il rischio di swing trading. Lo swing trading è per descrizione uno stile di investimento a lungo termine, quindi hai bisogno di un margine extra per gestire la natura esplosiva del mercato.

4. Informazioni sul tuo profilo:

Una delle prime e più importanti cose da fare quando inizi a fare trading in

il mercato è capire la tua avversione al rischio e volatilità. In altre parole, in quale fase della perdita un trader inizierà a farsi prendere dal panico? Se un trader swing ha un saldo del conto di 25.000€ e ha perso 3.000€, significa che ha perso il 10% del suo capitale. In tal caso, sarebbe un fallimento o penserebbe che sia normale? Il modo in cui il trader risponderà a questa perdita influenzerà i rischi che è disposto a correre nel trading.

Puoi vedere che questa strategia sarà facile da capire per un principiante. Pertanto, una strategia di Swing Trading è una strategia di trading a medio e lungo termine. Questa strategia dipende molto dal capitale e dalla gestione dei rischi. È più comunemente chiamato swing trading di gestione del denaro.

Gestione del denaro nello swing trading:

Una volta che un trader ha compreso il quadro generale del trading, deve ancora gestire il suo rischio ogni giorno nel mercato. E un modo per fare trading con saggezza è gestire i tuoi soldi con successo. Potrebbe dare l'impressione di una domanda complessa, ma in realtà non deve esserlo.

Ad esempio: se un trader volesse mantenere il rischio totale del 6% del saldo del suo conto in modo da poter aprire sei operazioni, ciascuna rischiando l'1% del suo patrimonio.

Per questo motivo, un trader potrebbe perdere l'1% del suo capitale in sei diverse operazioni o un importo massimo di € 200 in ciascuna operazione se hai un conto di € 20.000. A quel punto, prima di prendere posizione, un trader dovrebbe essere consapevole del suo rischio massimo per una corretta gestione del suo patrimonio nello Swing Trading che sarà dell'1% o 200 euro. Per questo motivo, la tua posizione di stop-loss e neutralizzazione sarà determinata in precedenza da ciascuna posizione. E da lì, dovresti eseguire quante più azioni possibili senza superare la tua gestione del rischio.

Quel limite influenzerà quindi le tue azioni e chiuderai poiché l'operazione si sta avvicinando al limite di perdita, oppure perderai l'operazione perché l'asset sale e raggiunge il profitto target. E se un'operazione supera il punto di pareggio, a quel punto sviluppa un'operazione 'neutra', puoi aprire un nuovo sito, senza mettere in pericolo il tuo limite di rischio.

I migliori strumenti per lo swing trading:

Esiste una gamma di strumenti che un trader può utilizzare per migliorare le sue possibilità di successo durante l'esecuzione di strategie di swing trading.

Alcune gamme consigliate:

Matrice di correlazione: *La relazione tra coppie Forex, forniture o le guide alle azioni sono un elemento di analisi che consente al trading ragionevole di operare con fiducia in se stessi.*

Mini grafici:

Un mini strumento di creazione di grafici consente al trader di analizzare numerose unità di tempo su un singolo grafico. Ciò significa che non è necessario che il trader passi dal grafico swing W1 a D1 per entrare nel grafico H4 per scoprire il suo punto di ingresso

Informazioni sui simboli:

Allo stesso modo, questo indicatore di oscillazione Forex consente ai trader di comprendere su un singolo grafico i segnali di trading degli indicatori massimi utilizzati su otto diverse scale di intervallo.

Miniterminale:

Lo strumento mini terminale consente al trader di aprire un posto in Meta Trader in un secondo, ma poi di nuovo, consente anche allo swing trader di aprire operazioni relative al rischio in euro permanenti o in percentuale. In effetti, ti fornisce diverse informazioni associate al mercato azionario o alla coppia di valute in cui un trader può applicarlo, inclusa la tendenza attuale, la forza dei movimenti attuali e lo slancio attuale

Altri indicatori utili per gli swing trader includono:

Media mobile esponenziale

MACD

Oscillatore travolgente

Parabolic SAR

CCI

Ammiraglio Donchian

Quindi ecco che arriva la domanda che preoccupa i principianti, che da dove i principianti dello swing trading possono accedere a questi strumenti di swing trading? È facile per coloro che hanno una demo o un conto live con Admiral Markets. E in caso contrario, la buona notizia è che tutti i principianti possono facilmente accedere a questi totalmente gratuiti con Meta-Trader Supreme Edition.

Meta-Trader Supreme Edition è un plug-in assolutamente gratuito per MT4 e MT5 che contiene una gamma di funzionalità non convenzionali, come un indicatore

pacchetto con idee di trading di analisi tecnica fornite da Trading Central e 16 nuovi indicatori e mini terminali e mini grafici per contrassegnare il tuo trading ancora più efficace.

I migliori consigli per lo swing trading Forex per principianti:

Dopo aver conosciuto le basi dello swing e dopo aver avuto abbastanza informazioni sulle strategie di swing trading Forex, ecco i migliori consigli che ti aiuteranno ad avere successo come swing trader.

1. Abbina le tue operazioni alla tendenza a lungo termine:

Sebbene un trader possa sempre guardare un grafico temporale a breve termine (ad es. H1 o H4), può anche confortare il principiante guardare un grafico a lungo termine (D1 o W1) per ottenere informazioni sul lungo termine. andamento del termine. Quindi puoi assicurarti di non fare trading accanto a una tendenza più ampia. Lo swing trading è anche molto più semplice quando è stato scambiato per mezzo del trend, piuttosto che contro il trend.

2. Ottieni il meglio dalle medie mobili (MA):

L'indicatore MA può trarre profitto dal trader per classificare le tendenze moderando le differenze di prezzo a breve termine. Poiché l'MA comprende i vecchi prezzi dei dati, è il modo più semplice per associare il modo in cui i prezzi esistenti si collegano ai prezzi precedenti.

3. Usa una piccola leva:

La leva consente al trader di accedere a una posizione più ampia rispetto a quella che normalmente consentirebbe al tuo deposito, oltre a rafforzare i tuoi profitti e le tue perdite. Una volta usata saggiamente, la leva finanziaria può aiutare il trader a ottenere il massimo dalle operazioni vincenti.

4. Fai trading su un ampio portafoglio di coppie Forex:

Cerca più coppie di valute che puoi per scoprire le migliori opportunità. Il mercato Forex ti offrirà ogni volta le opportunità di trading che desideri; devi cercare quelli che corrispondono meglio al tuo segnale. Inoltre, il trading di una gamma di coppie ti aiuterà a diversificare il tuo portafoglio e a correre il rischio di avere tutte le uova nello stesso paniere.

5. Presta attenzione agli swap:

Gli swap sono un costo di negoziazione addebitato pronto per le posizioni detenute durante la notte. Questi swap devono essere presi in considerazione nello swing trading del trader per poter gestire meglio i tuoi soldi.

6. Mantenimento di un rapporto utile/perdita positivo:

Che si tratti di H4 o di trading giornaliero, lo swing trading consente al trader di attingere a grandi pianificazioni di mercato, dando al trader la possibilità di ottenere rapporti di profitto maggiori associati a perdite molto piccole possibili, soprattutto se abbinato allo scalping.

7. Metti da parte i tuoi sentimenti:

È meglio fare trading senza emozioni, ma poi di nuovo, fare scambi swing come parte di un piano e di una strategia di trading Forex fissi.

Scegli un broker per Forex Swing:

Prima che un trader inizi a fare trading, un trader deve scegliere un broker. La scelta di un broker per il mercato Forex darà la possibilità ai nuovi trader di accedere ai mercati nel modo in cui vogliono fare trading, insieme a una piattaforma di trading per svolgere le proprie operazioni. Sebbene alcuni broker siano migliori di altri, ecco perché è essenziale tenere a mente quanto segue mentre si effettua una scelta.

Controlla se sono sincronizzati dal regolatore locale nella tua zona? Ammiraglio Markets è un broker Forex e CFD controllato da EFSA, ASIC CYSEC e FCA.

I costi di negoziazione contengono lo spread, lo swap e gli ordini sulle negoziazioni, che fanno come mangiare nei tuoi profitti. Ecco perché è essenziale conoscere i costi di trading tipici.

Una quota Forex standard, o contratto di trading, vale 100.000 della base valuta della coppia prima della prima valuta quotata (se un'azione di EUR/USD costa EUR 100.000). Per i nuovi trader, questo potrebbe essere più di quello che vuoi vedere, quindi controlla se il broker tratta lotti micro (0,01) e mini (0,1) per il trading.

Per prendere decisioni di trading consapevoli, è essenziale avere le ultime novità informazioni di mercato. I buoni broker Forex e CFD si occuperanno dei prezzi in tempo reale nella loro piattaforma di trading.

Il punto successivo è conoscere il mercato e quanta leva finanziaria il broker offre? Se parli di Europa, in Europa, i broker

dovrebbe offrire l'accesso alla leva finanziaria fino a 1:500 per i clienti specializzati e 1:30 per i clienti al dettaglio.

Qual è l'importo minimo di cui un trader ha bisogno per iniziare a fare trading? All'ammiraglio Markets, un trader può fornire al conto di trading un minimo di € 100. Ciò consente inoltre al trader di avviare piccoli scambi senza correre grossi rischi e aggiungere man mano che si impara la psicologia del mercato e il comportamento del trading autogovernato.

Admiral Markets può aiutare il trader a ridurre il suo rischio di trading con difesa dalla volatilità e protezione dal saldo negativo.

Will, il broker, consente al trader non semplicemente di oscillare il commercio, ma il commercio diurno e anche lo scalpo, se fa parte della strategia?

Il broker nello swing trading fornisce strumenti e risorse per aiutare il? principiante per avere successo come trader? Come se parlassimo di Admiral Markets, ad esempio, ha una libreria di più di centinaia di articoli Forex, corsi gratuiti come [Forex 101](#), e libero [webinar di trading](#).

Strategie di Swing Trading Forex

Un sommario

Lo swing trading è uno stile adatto a mercati volatili e suggerisce anche frequenti possibilità di trading. Sebbene il trader abbia bisogno di capitalizzare un ragionevole lasso di tempo per osservare il mercato con lo swing trading, le forniture non sono così difficili come gli stili di trading con intervalli di tempo più brevi, come il day trading o lo scalping. Nel calcolo, anche se dai il favore al day trading o allo scalping, lo swing trading ti offrirà poche diversificazioni nei tuoi risultati e profitti aggiuntivi. Si dice che lo swing trading non sia per tutti i trader, quindi è meglio praticarlo prima senza rischi con un conto di trading demo.

Mercati dell'ammiraglio:

Admiral Markets è un mercato che ha ottenuto molti successi nel mercato grazie al suo pluripremiato risultato. Broker Forex e CFD, che fornisce trading su oltre 8.000 strumenti finanziari tramite le piattaforme di trading più famose al mondo:

1.MetaTrader 4

2.MetaTrader 5

Questo sostanziale non regge e non deve essere interpretato come contenente consigli sugli asset, raccomandazioni sugli asset, un'offerta o una sollecitazione per qualsiasi operazione in strumenti finanziari. Un principiante non dovrebbe dimenticare che tale analisi di trading non è un indicatore affidabile per qualsiasi performance attuale o futura, poiché le condizioni possono cambiare nel tempo. Prima di trarre conclusioni sugli investimenti, dovrebbe chiedere consiglio a consulenti finanziari indipendenti per confermare di aver compreso il [rischi](#).

4. INDICATORI SWING E DAY TRADING



Quali sono gli indicatori di trading?

"Gli indicatori di trading sono calcoli matematici progettati per prevedere quale sarà il mercato fare."

T gli indicatori di rating sono tracciati come linee su un grafico dei prezzi e possono confortare i trader nell'identificare determinati segnali e tendenze all'interno del mercato. L'indicatore numero uno potrebbe essere un segnale di previsione che calcola i movimenti futuri dei prezzi, mentre un indicatore in ritardo esamina le tendenze passate e indica lo slancio.

Perché gli indicatori tecnici sono importanti?

Gli indicatori tecnici sono algoritmi supportati che praticano i dati di prezzo precedenti nel calcolo. Di conseguenza, tutti gli indicatori tecnici di trading sono in ritardo nel loro ambiente naturale, ma ciò non significa che non possano restituire informazioni utili una volta al giorno facendo trading sui mercati. Privati dell'assistenza di indicatori, i trader avrebbero difficoltà calcolare questa volatilità dei mercati, la forza di una tendenza o se le condizioni di mercato sono ipercomprate o ipervendute.

Detto questo, un'intera strategia di trading non dovrebbe dipendere esclusivamente da indicatori tecnici. Restituiscono i risultati più semplici come strumento di conferma. Non acquistare solo perché l'RSI è inferiore a 30 o vendere perché l'oscillatore stocastico aumenta direttamente sopra 80. In alternativa, un trader dovrebbe creare una strategia di trading definita (costruita sull'azione dei prezzi o sui fondamentali, per esempio) e utilizzando tecniche indicatori semplicemente per convalidare una possibile configurazione e modificare i livelli di ingresso.

Tipi di indicatori tecnici

A seconda delle conoscenze fornite dagli indicatori tecnici, verranno raggruppati in tre categorie principali:

Indicatori di tendenza.

Indicatori di slancio.

Indicatori di volatilità.

1. Indicatori di tendenza

Gli indicatori trend-follower sono abituati a determinare le tendenze e a vivere il punto di forza di un mercato in trend. Mentre la maggior parte dei trader è in grado di identificare una tendenza semplicemente fissando il grafico del valore, è spesso difficile vivere la sua forza o identificare una tendenza all'inizio della sua creazione. Gli indicatori di tendenza comuni contengono anche medie mobili, MACD e anche l'indicatore ADX, per citarne alcuni.

2. Indicatore di momento:

Di solito misura la forza dei recenti movimenti di prezzo rispetto ai periodi precedenti. Variano tra 0 e 100, a condizione dei segnali dell'indicatore di condizioni di mercato ipercomprato e ipervenduto. Gli indicatori di momentum restituiscono un segnale di marketing quando i valori iniziano a muoversi molto più in alto e un segnale di acquisto quando i prezzi iniziano a muoversi molto più in basso. Mentre questo sarà redditizio nei mercati che vanno, gli indicatori di momentum di solito restituiscono falsi segnali durante le tendenze forti. Alcuni esempi di indicatori di momentum includono RSI, Stochastics e CCI.

3. Indicatori di volatilità:

Indicatori di volatilità, come suggerisce il loro nome, misura la volatilità dello strumento fondamentale. I trader generalmente inseguono la volatilità da un angolo all'altro dei mercati d'angolo per scovare opportunità di trading redditizie, il che rende gli indicatori di volatilità uno strumento forte per il day trading. Esempi di indicatori di volatilità includono Bollinger Bands e anche l'indicatore ATR, tra gli altri.

Ogni investitore ha fiducia nella strategia di acquisto e detenzione. L'unico argomento di discussione è quanto tempo dovrebbe durare l'era dell'holding. Per ogni adolescente che ha iniziato a comprare azioni non apprezzate e le detiene per otto decenni, raccogliendo azioni lungo la strada, ci sono dozzine di altri che si prendono dei rischi che non vedono l'ora di uscire dalle loro posizioni in meno di una settimana. Ciò non solo richiede che un titolo salga rapidamente, ma anche che salga di prezzo in modo tale da compensare eventuali costi di materia. Lo swing trading è per l'azionista che di fatto

non vedo l'ora che arrivi il fine settimana.

Lo swing trading è il processo di trading più veloce e gestibile da tutti, anche dai principianti che sono appena entrati nel mondo del trading. La velocità dello swing trading è più lenta del day trading, il che fornisce anche al trader il tempo sufficiente per formulare una pratica ed eseguire una piccola ricerca prima di prendere decisioni sul tuo trade. Lo swing trading è anche un metodo raro per coloro che desiderano creare un'incursione nel day trading per aumentare le proprie capacità prima di intraprendere il processo di day trading più complesso.

Questi indicatori tecnici sono gli strumenti matematici che possono fornire informazioni fruibili da un piano azionario che a volte può sembrare disinformato. Nelle mani di un acquirente di rischi adeguatamente fortunato, i giusti indicatori tecnici possono significare una possibilità di profitto. Ora, solo una rara tra quelle più comunemente usate dagli swing trader, in ordine crescente di complessità.

Gli swing trader tendono a non avere gli stessi obiettivi dei day trader, che mirano a raccogliere rapidi cambiamenti intraday a causa di un catalizzatore.

Gli scambi swing, d'altra parte, tendono a puntare a "swing" che passano da un minimo a breve termine a un nuovo massimo.

I segnali di swing trading tendono a non sembrare uguali ai segnali di day trading. I day trader cercano spesso volatilità e volume elevati e cercano di cavalcare una tendenza, possibilmente acquistando a un pullback a VWAP.

Prima di parlare degli indicatori di trading, è fondamentale capire cosa rappresentano e perché e come lo rappresentano sono anche domande importanti su cui riflettere. Gli indicatori di swing trading non sono migliori di qualsiasi altro metodo di analisi tecnica, e certamente non avrebbero dovuto essere visti come il graal divino. Non è garantito che qualsiasi operazione effettuata da un trader produrrà profitto solo per il motivo per cui un indicatore di trading lo ha segnalato.

Ecco alcuni altri fattori che possono influenzare la creazione di un profitto o una perdita nello swing trading:

Le condizioni di mercato possono spesso ridurre l'efficacia degli indicatori. Persino se sembra che la sicurezza sia vicina a salire, un'ampia emozione ribassista potrebbe causare un'ulteriore caduta di valore.

I tempi del tuo commercio devono essere rigorosi. Ricevere il momento giusto può essere autorevole per creare un enorme profitto.

Ciò che molti chiamano percezione è semplicemente know-how. Come tutte le altre cose, diventerai un trader migliore mentre passi più tempo a fare trading e

cogliere sottili segnali di mercato.

Indicatori dello swing trading:

Gli indicatori di swing trade sono importanti su cui concentrarsi quando si seleziona quando acquistare, quando fare trading e cosa comprare. Di seguito sono riportate alcune delle migliori combinazioni di indicatori per lo swing trading.

1. Medie mobili:

La media mobile è uno dei principali indicatori di trading di base nello swing trading. Una media mobile attenua i movimenti irregolari dei prezzi a breve termine e ci conforta per comprendere meglio la tendenza e in che modo si sta muovendo la sicurezza. Le medie mobili sono buoni indicatori di per sé, ma vengono anche utilizzate come base per altri indicatori più descrittivi. Lo swing trader non dovrebbe dimenticare che le medie mobili sono sempre in ritardo rispetto al prezzo attuale grazie al factoring nei dati passati. Più dati pensi, maggiore è il ritardo. Pertanto, come un trader swing. È intelligente mescolare le medie mobili a breve termine con le medie mobili a lungo termine. Così facendo, pensi sia al trend di lungo che a quello a breve termine e hai più terreno su cui basare le tue decisioni.

Quando un trader sta osservando le medie mobili, osserverà le linee calcolate costruite sui prezzi passati. Questo indicatore non è difficile da capire, ed è anche difficile capire se stai facendo day trading, swing trading o anche trading a lungo termine. Sono utilizzati per confermare una tendenza o identificare una tendenza. Per decidere la media, un trader dovrà sommare tutti i prezzi finali e il numero di giorni coperti dal periodo e quindi dividere i prezzi finali per il numero di giorni. Per utilizzare con successo le medie mobili, un trader dovrà calcolare diversi periodi di tempo e collegarli su un grafico. Questo darà al trader una visione più ampia del mercato, così come le loro variazioni medie nel tempo. Quando hai pianificato le tue medie mobili, devi usarle per prendere in considerazione le tue decisioni commerciali.

Riconosci la forza di una tendenza:

Se il prezzo attuale del titolo e del trend è al di là della sua media mobile, allora è considerato un trend più debole. La forza del trend, condivisa da un indicatore come il volume, può aiutarti a creare scelte migliori sulle tue operazioni.

Determinazione delle inversioni di tendenza:

È possibile utilizzare le medie mobili per riconoscere le inversioni di tendenza con limiti. È necessario osservare i casi in cui le medie mobili correnti incrociano le medie mobili più lunghe dopo un trend rialzista. Sebbene questo non sia l'unico strumento, dovresti usarlo per regolare l'inversione, ma può confortarti nel determinare se dovresti esplorarlo ulteriormente.

2. Indice di forza relativa:

Welles Wilder ha sviluppato l'indicatore RSI negli anni '70 e ha distribuito le sue scoperte in *New Concepts in Technical Trading Systems*. Il libro potrebbe essere un testo classico che presentava più indicatori tecnici convenzionali come l'RSI, l'Average True Range e, quindi, l'indice direzionale medio. Una semplice spiegazione del calcolo di RSI è collegare questa forza dei prezzi rispetto alla forza dei prezzi passata. Ad esempio, un RSI a 14 periodi su un grafico giornaliero confronterà il prezzo di oggi con le ultime 13 chiusure. Una lettura alta indica che il prezzo di oggi è potente, rispetto alle precedenti 13 chiusure, e viceversa. Wilder aveva pianificato che l'RSI potesse essere usato come un oscillatore di momentum: misurando esattamente quanto forte fosse il momentum durante un mercato, ma poi l'RSI è stato utilizzato in un modo diverso. L'RSI è utilizzato principalmente dai trader come indicatore di ipercomprato-ipervenduto, dove un'ottima lettura significa che il titolo è "ipercomprato" ed è inevitabile un pullback. Nel confronto, una lettura "ipervenduto" indica che il mercato è in attesa di un rally perché è stato venduto troppo. Questo consumo dell'indicatore si confronta con l'obiettivo di un oscillatore di slancio, contenuto dal quale un'ottima analisi indica che questa tendenza è ulteriormente destinata a durare. Nelle serie temporali convenzionali di RSI a 14 periodi viene utilizzato. La situazione è quella suggerita anche nel lavoro di Wilders ed è inadempiente nelle piattaforme di massima creazione di grafici. D'altra parte, lo studio del punto di Larry Connors verso l'RSI a 14 periodi comprende un bordo in miniatura, in cui le analisi RSI a breve termine producono segnali commerciali aggiuntivi. dove un'ottima lettura significa che il titolo è "ipercomprato" ed è inevitabile un pullback. Nel confronto, una lettura "ipervenduto" indica che il mercato è in attesa di un rally perché è stato venduto troppo. Questo consumo dell'indicatore si confronta con l'obiettivo di un oscillatore di slancio, contenuto dal quale un'ottima analisi indica che questa tendenza è ulteriormente destinata a durare. Nelle serie temporali convenzionali di RSI a 14 periodi viene utilizzato. La situazione è quella suggerita anche nel lavoro di Wilders ed è inadempiente nelle piattaforme di massima creazione di grafici. D'altra parte, lo studio del punto di Larry Connors verso l'RSI a 14 periodi comprende un bordo in miniatura, in cui le analisi RSI a breve termine producono segnali commerciali aggiuntivi. dove un'ottima lettura significa che il titolo è "ipercomprato" ed è inevitabile un pullback. Nel confronto, una lettura "ipervenduto" indica che il mercato è in attesa di un rally perché è stato venduto troppo. Questo consumo dell'indicatore si confronta con l'obiettivo di un oscillatore di slancio, contenuto dal quale un'ottima analisi indica che questa tendenza è ulteriormente destinata a durare. Nelle serie temporali convenzionali di RSI a 14 periodi viene utilizzato. La situazione è quella suggerita anche nel lavoro di Wilders ed è inadempiente nelle piattaforme di massima creazione di grafici. D'altra parte, lo studio del punto di Larry Connors verso l'RSI a 14 periodi comprende un bordo in miniatura, in cui le analisi RSI a breve termine producono segnali commerciali aggiuntivi. una lettura "ipervenduto" indica che il mercato è in attesa di un rally perché è stato venduto troppo. Questo consumo dell'indicatore si confronta con l'obiettivo di un oscillatore di slancio, contenuto dal quale un'ottima analisi indica che questa tendenza è ulteriormente destinata a durare. Nelle serie temporali convenzionali di RSI a 14 periodi viene utilizzato. La situazione è quella suggerita anche nel lavoro di Wilders ed è inadempiente nelle piattaforme di massima creazione di grafici. D'altra parte, lo studio del punto di Larry Connors verso l'RSI a 14 periodi comprende un bordo in miniatura, in cui le analisi RSI a breve termine producono segnali commerciali aggiuntivi.

La ricerca di Connors indica che i periodi RSI 2, quattro e misto mostrano l'aspettativa commerciale a lungo termine più efficace. Come risultato della sua ricerca, Connors ha sviluppato l'indicatore Connors RSI (che misura la velocità di cambiamento di RSI), che è incluso nella maggior parte dei pacchetti di grafici al giorno d'oggi.

L'indice di forza relativa (RSI) è uno dei migliori indicatori tecnici per lo swing trading. Questo indicatore è responsabile di fornire le informazioni necessarie per determinare una volta che l'ideale entra nel mercato. Consente ai trader di

indagare meglio i piccoli segnali. Ciò aiuterà il trader a regolare se il mercato è stato ipervenduto o ipercomprato, è range-bound o è piatto. L'RSI fornirà al trader una valutazione relativa su come garantire il prezzo esistente esaminando sia l'instabilità che le prestazioni passate. Questo indicatore sarà facilmente riconoscibile utilizzando un intervallo di 1-100. L'indicatore RSI è più conveniente per:

Determinare le circostanze che hanno portato il mercato all'ipervenduto o ipercomprato:

Un trader dovrà essere in grado di classificare queste condizioni in modo che un trader possa trovare ugualmente inversione di tendenza e correzioni. L'overbuying può indicare una tendenza ribassista anche se l'overselling può essere visto come più rialzista. Se l'indicatore è intorno a 30, potrebbe essere indicato come un sottovalore o un ipervenduto. Indicatori circa 70 possono significare che il titolo è stato sopravvalutato o ipercomprato.

Identificazione delle divergenze:

Le divergenze sono utilizzate per classificare le inversioni nelle tendenze. Quando il valore raggiunge un nuovo minimo, ma l'RSI no, allora in quel momento sarebbe considerato un segnale divergente rialzista. Se il valore raggiunge un nuovo massimo e l'RSI no, allora verrebbe definito un segnale ribassista.

3. Volume:

Un indicatore generalmente trascurato che è più facile da usare per i principianti, anche per i nuovi trader, è il volume. Guardare il volume è molto critico quando si considerano le tendenze. Le tendenze devono essere mantenute e supportate dal volume. Un trader vuole sempre assicurarsi che ci sia un volume più significativo in corso quando la tendenza sta andando in quel modo. Aumentare il volume significa denaro a sostegno della sicurezza, e se non vedi il volume, potrebbe essere un'indicazione che ci sono condizioni sottovalutate in gioco.

4. Indicatore di analisi visiva:

Nonostante il fatto che gli indicatori tecnici per lo swing trading siano fondamentali per fare le scelte giuste, è utile per molti azionisti, sia nuovi che esperti, essere in grado di guardare i modelli visivi. Generando modelli visivi, un trader può vedere le attività sul mercato con una rapida occhiata per aiutare a supportare la tua decisione.

5. Il flusso degli squilibri netti:

Ogni giorno, ci sono molti ordini da trovare per sollecitare la stampa di chiusura (ordini di mercato alla chiusura). Si tratta in genere di istituzioni come i fondi comuni di investimento

ed ETF che necessitano della grande liquidità fornita alla chiusura del mercato. Ogni giorno, qualche minuto prima della chiusura, le interazioni distribuiranno informazioni sulle disuguaglianze degli ordini alla chiusura del mercato. Cioè, quale numero di azioni vengono accettate in più rispetto a quelle vendute alla chiusura? Ad esempio, se gli ordini di acquisto di mercato alla chiusura sono pari a 100.000 azioni e gli ordini di vendita pari a 90.000 azioni, si tratta di uno squilibrio di +10.000 azioni.

I market maker lo arbitrano nel breve termine e guadagnano centesimi, non è un lasso di tempo in cui siamo in grado di competere. Tuttavia, siamo in grado di monitorare il flusso di cassa di un titolo o di un settore monitorando l'andamento degli squilibri online nel tempo.

Se ci sono determinati a chiudere gli squilibri in un settore, significa che le istituzioni stanno raccogliendo un vantaggio in esso. Questo ci fornisce informazioni vitali sul fatto che un'importante mossa di prezzo potrebbe essere sul precipizio. Lo strumento che sfrutto per misurare lo squilibrio del flusso di denaro è Market Chameleon. MC consente di visualizzare facilmente le medie mobili di 20 o 50 giorni degli afflussi e deflussi di capitali di settori, industrie, indici o liste di controllo.

Indicatori di trading giornaliero:

Per trovare gli indicatori tecnici più semplici per il tuo particolare approccio al day trading, provane alcuni singolarmente e così insieme. Ti ritroverai ad attaccare, diciamo, quattro che sono sempreverdi, altrimenti potresti capovolgere il conto sull'asset che stai negoziando o sulle situazioni di mercato del giorno.

Indipendentemente dal fatto che tu stia negoziando azioni giornaliere, forex o futures, spesso è meglio rimanere sul semplice quando si tratta di indicatori tecnici. Scoprirai che ti piace visualizzare solo un paio di indicatori per approvare i punti di ingresso e i punti di uscita. Al massimo, usa semplicemente uno per ogni tipo di indicatore per evitare ripetizioni evitabili e che distraggono.

Combinazione di indicatori di day trading

Considera una combinazione di insiemi di due indicatori sul grafico delle spese per classificare i punti da avviare e ottenere da un'operazione.

Ad esempio, l'indice di forza relativa RSI e la convergenza/divergenza della media mobile verranno combinati sullo schermo per consigliare e rafforzare un segnale di trading.

L'indice di forza relativa (RSI) può consigliare condizioni di ipercomprato o ipervenduto misurando il momentum del valore di un asset. L'indicatore era

prodotto da J. Welles Wilder Jr, che ha proposto che lo slancio che raggiunge il 30 (su una scala da zero a 100) fosse un'indicazione di un'attività in ipervenduto, quindi una possibilità di acquisto e un livello del 70 per cento erano un'indicazione di un'attività in ipercomprato e quindi una possibilità di vendita o di vendita allo scoperto.

Constance Brown, CMT, ha perfezionato il servizio dell'indice e ha ipotizzato che il livello di ipervenduto in un mercato con tendenza al rialzo fosse fondamentalmente molto al di sopra di 30, e quindi, il livello di ipercomprato in un mercato con tendenza al ribasso era molto inferiore a 70,3. Utilizzando i livelli di Wilder, il prezzo dell'asset può ancora aumentare per alcune volte, mentre l'RSI indica ipercomprato e viceversa. Per questo motivo, l'RSI è meglio monitorato solo se il suo segnale imita il trend del valore: ad esempio, cerca indicazioni di momentum ribassista quando il trend del valore è ribassista e non prestare attenzione a quelle indicazioni quando il trend del valore è rialzista.

Per identificare più facilmente tali tendenze di prezzo, sarai in grado di utilizzare l'indicatore di convergenza/divergenza media mobile (MACD). MACD è costituito da due linee del grafico. La linea MACD viene creata eliminando una media mobile esponenziale a 26 periodi (EMA) da una EMA a 12 periodi. Un EMA è il valore medio di un bene in un periodo del tuo tempo solo con la modifica chiave che ai prezzi più recenti viene data una maggiore tolleranza rispetto ai prezzi più lontani.

La seconda linea è la linea del segnale e potrebbe essere una EMA a 9 periodi. Una tendenza ribassista viene segnalata una volta che la linea MACD attraversa la linea del segnale; un trend rialzista viene segnalato dopo che la linea MACD ha incrociato direttamente sopra la linea di segnale.

Scelta delle coppie:

Durante la selezione delle coppie, è una buona conoscenza decidere su un indicatore che ha misurato un indicatore numero uno, ad esempio l'RSI, e uno che è un indicatore in ritardo, ad esempio, come MACD. Gli indicatori più importanti emettono segnali prima che si creino le condizioni per entrare nel commercio. Gli indicatori in ritardo generano segnali di avvertimento dopo che tali circostanze sono apparse, in modo che possano fungere da verifica degli indicatori principali e potrebbero impedirti di fare trading su segnali inventati.

Un trader deve anche dare un voto di prim'ordine a un abbinamento che presenta indicatori di due dei quattro diversi tipi, mai due dello stesso tipo. I quattro tipi sono sviluppo come MACD, momenti come RSI, volatilità e volume. Come suggeriscono i loro nomi, gli indicatori di volatilità sono la volatilità supportata all'interno del prezzo dell'asset e gli indicatori di volume sono i volumi di trading supportati dell'asset. In genere non è utile guardare due indicatori dello stesso tipo

poiché saranno a condizione che le informazioni identiche.

Utilizzo di più indicatori:

Un trader potrebbe anche scegliere di avere un indicatore live di ogni tipo; potrebbero essere due dei quali sono al massimo sostanziali e due dei quali sono in ritardo. Numerosi indicatori possono fornire un rafforzamento ancora maggiore dei segnali di trading e potrebbero aumentare le probabilità di dare la caccia a segnali inventati.

Indicatori di raffinamento:

Qualunque siano gli indicatori che scegli, assicurati di ricercarli e di fare un riepilogo della loro efficacia nel tempo. Chiediti: quali sono gli svantaggi di un indicatore? Produce diversi segnali inventati? Non riesce a segnalare, portando a occasioni mancate? Segnala troppo presto (più probabile di un indicatore numero uno) o troppo tardi (più probabile di uno in ritardo)?

Potresti scoprire che un indicatore è effettivo quando fai trading di azioni ma non, ad esempio, forex. Potresti voler sostituire un indicatore con uno aggiuntivo del suo tipo o apportare modifiche al modo in cui è pianificato. Apportare tali modifiche potrebbe essere una parte fondamentale del successo quando si fa trading giornaliero con indicatori tecnici.

Dovresti fare trading su indicatori tecnici?

Gli indicatori tecnici praticano i dati sui prezzi passati nel loro calcolo e, di conseguenza, ritardano questo prezzo. D'altra parte, poiché i dati storici sono l'unica conoscenza che i trader devono fare in anticipo per i futuri movimenti di prezzo, gli indicatori tecnici hanno un ruolo molto importante durante una strategia di trading ben definita.

Evitare l'aggiunta di troppi indicatori al grafico poiché indicatori dello stesso tipo generalmente restituiscono segnali di trading simili. In sostituzione di, scegli solo un indicatore da ogni gruppo (momentum, trend following e volatilità) e mescola le loro indicazioni per verificare una situazione e il commercio l'ha supportata. Una combinazione efficiente di indicatori potrebbe essere il movimento intorno, l'indicatore RSI e, quindi, l'indicatore ATR, per esempio.

Non basare le tue decisioni di trading totalmente sugli indicatori e sui loro segnali. Gli indicatori che seguono il trend restituiscono un segnale di acquisto quando i prezzi iniziano a muoversi verso l'alto, indipendentemente dal fatto che il mercato si sposti lateralmente o meno. Nello stesso

modo, oscillatori e indicatori di momentum ti offriranno un segnale di marketing quando i prezzi iniziano ad aumentare durante un trend rialzista. Non esiste un unico indicatore più grande, motivo per cui dovresti combinare diversi tipi di indicatori e includerli in una strategia di trading più ampia.

5. PRO E CONTRO DI SWING E DAY COMMERCIO

Swing trading è di gran lunga uno dei modi più popolari per commerciare sui mercati commerciali. Ma come ogni stile di strategia, ci sono sia pro che contro quando lo usi, e conoscere quei tempi precedenti può essere cruciale per scegliere se è per te a lungo termine.

Vantaggi dello Swing Trading

Ti consente di richiedere il beneficio del naturale flusso e riflusso dei mercati.

I mercati finanziari non vanno mai in una direzione continuamente, e avendo la capacità di richiedere il beneficio di ciò, aumenterai i tuoi rendimenti mentre, in teoria, stai facendo soldi una volta che il mercato sale nei giorni successivi, quindi guadagnerai un po' di tempo il mercato si tira indietro, perché prima o poi lo farà sicuramente.

In realtà dentro e fuori i mercati, identificherai più possibilità. Se tu studiando qualsiasi grafico economico, vedrai che c'è quasi sempre una precisa tendenza a lungo termine, ma il mercato potrebbe non essere continuamente in un'area di sostentamento o resistenza. Essendo in un mercato estremamente e fuori dal mercato nel giro di alcuni giorni (in genere), raccoglierai profitti e identificherai altri mercati che stanno effettuando altre operazioni. Ciò ti consente di diffondere il pericolo in giro e impegnare molto meno capitale piuttosto che dover continuamente recuperare un margine per nuove posizioni di zecca mentre scopri nuovi mestieri. Chiudendo la tua prima posizione, non depositerai denaro extra sul tuo conto per nascondere la seconda.

Gli stop loss sono in genere più piccoli delle operazioni di lungo periodo. Gli stop loss su a

Lo swing trade potrebbe essere di 100 pips basato su un grafico a quattro ore, mentre uno stop loss su un grafico settimanale basato sulla tendenza potrebbe dover essere di 400 pips. Ciò ti consente di inserire posizioni di dimensioni maggiori anziché quelle con leva finanziaria estremamente bassa tramite le tendenze a lungo termine.

Hai confini chiari. Lo swing trader potrebbe essere un tecnico in più trader basato, e di per sé avrà normalmente un'area particolare che ritengono essere un'indicazione che il commercio sta funzionando contro di loro. Dovuto a

questo, riconosci esattamente quando lo scambio non funziona e può limitare il danno che un cattivo scambio può fare. I trader a lungo termine normalmente devono fornire un ampio ormeggio ai mercati in quanto non vedono l'ora che "seguano i fondamentali".

Svantaggi dello Swing Trading

Verrai segato spesso, semplicemente perché il mercato mostra supporto o resistenza in una particolare area, non significa che la rispetteranno oggi. Inoltre, ogni volta che puoi effettuare uno scambio, stai rischiando denaro. Per questo motivo, come swing trader, stai rischiando più spesso. Le probabilità sono che di tanto in tanto avrai delle perdite, indipendentemente da quanto sei bravo.

Imparerai a conoscere l'analisi tecnica. considerando che non necessariamente uno "svantaggio", significa lavoro extra. Quasi chiunque può dire la tendenza su un grafico che sta andando da in basso a sinistra a in alto a destra nel tempo, ma qualcuno stava cercando di far oscillare il commercio che un grafico deve identificare i punti di entrata e di uscita. Questo è spesso qualcosa che un'analisi tecnica può fare, ma dovresti prima parlarne. Questo richiede tempo.

Ci vuole una mentalità unica rispetto al trading a lungo termine e più nervi saldi. mentre è non è necessariamente scalping, lo swing trader corre il pericolo di essere "allarmato dai mercati" poiché i pullback in questi intervalli minori sembrano essere più violenti rispetto a qualcuno che osserva un grafico settimanale. Questo è spesso un problema psicologico che la maggior parte dei trader alla fine risolverà durante la propria carriera.

Come vedrai, ci sono pro e contro nello swing trading, un po' come il resto. Ad essere onesti, la maggior parte dei trader fa un tocco di stili diversi perché i mercati non sono necessariamente sempre favorevoli ad almeno un particolare stile di trading e talvolta possono coinvolgerne altri. Un trader onesto è in grado di utilizzare vari stili di trading in modo da estendere i propri fondi. Il trader deve adattarsi ai mercati, non il contrario.

Esempio di swing trading:

Trova un titolo nel mercato che è stato scambiato nella direzione del rialzo nell'ultima settimana e ha preparato minimi brevi e taglienti sul suo grafico giornaliero. Inoltre, determina la performance del titolo dall'inizio del trend rialzista e nota se il suo prezzo è stato rimborsato alla media mobile tre volte. Se lo facesse, e penetrasse anche nella media mobile a una mediana dell'1,5% del suo prezzo, potrebbe essere piazzato un buon ordine di acquisto. Questo ordine può essere circa l'1% delle scorte

prezzo al di sotto della media mobile.

Quando il trade si è mosso, è consigliabile mettere uno stop loss vicino al punto di ingresso per frenare le perdite. E i profitti possono essere presi vicino alla linea del canale superiore per i mercati deboli e alla linea della stazione superiore, per i mercati forti. Lo scopo è quello di ottenere profitti in linea con il tuo piano di trading, ma un trader esperto può preferire tenere per anni più a lungo fino a quando il mercato non raggiunge nuovi massimi.

Vantaggi del Day Trading:

Strategia di trading:

Il day trading ti consente di utilizzare una gamma di strategie di trading in tutti i principali mercati. Le comuni strategie di day trading includono il breakout trading, il controtrend trading e il trend following (o mean-reversion). Nel trading breakout, i trader cercano di catturare la volatilità iniziale che si verifica istantaneamente dopo che il valore supera un livello tecnico molto importante, come i modelli grafici. Gli ordini incompleti tendono a raggrupparsi sopra e sotto livelli importanti, il che finisce in un flusso di slancio e volatilità dopo che il valore raggiunge quei livelli e innesca gli ordini indecisi. Il breakout trading consente inoltre ai day trader di allineare un ordine incompleto per catturare un breakout una volta che si verifica, poiché gli ordini in sospeso diventano ordini di mercato una volta che il valore raggiunge il livello prestabilito. Gli strumenti tecnici più diffusi utilizzati dai trader di breakout contengono modelli grafici, come modelli testa e spalle, triangoli, doppi massimi e minimi, tripli massimi e minimi, rettangoli, cunei e bandiere. Inoltre, i trader di breakout possono anche sfruttare al meglio la volatilità che si verifica dopo che il valore si rompe sopra o sotto un canale, una linea di tendenza o un finanziamento orizzontale o livelli di resistenza. Le strategie che seguono il trend, come suggerisce il nome, includono le negoziazioni del giorno di apertura nella direzione della tendenza intraday sottostante. Il trend-following è probabilmente la strategia di trading più popolare tra i day trader perché restituisce un ottimo rapporto rischio/rendimento con un tasso di successo relativamente alto. Per aprire lo scambio sulla via del trend sottostante, attendi il valore per terminare un pullback (ad esempio, a un livello di Fibonacci intraday molto importante) e usa i modelli di candele per assicurarti che il trend sottostante sia sul punto di continuare. triple superiori e inferiori, rettangoli, cunei e bandiere. Inoltre, i trader di breakout possono anche sfruttare al meglio la volatilità che si verifica dopo che il valore si rompe al di sopra o al di sotto di un canale, una linea di tendenza o un finanziamento orizzontale o livelli di resistenza. Le strategie che seguono il trend, come suggerisce il nome, includono le negoziazioni del giorno di apertura nella direzione della tendenza intraday sottostante. Il trend-following è probabilmente la strategia di trading più popolare tra i day trader perché restituisce un ottimo rapporto rischio/rendimento con un tasso di successo relativamente alto. Per aprire lo scambio sulla via del trend sottostante, attendi il valore per terminare un pullback (ad esempio, a un livello di Fibonacci intraday molto importante) e usa i modelli di candele per assicurarti che il trend sottostante sia sul punto di continuare. i trader di breakout possono anche sfruttare al meglio la volatilità che si verifica dopo che il valore si rompe sopra o sotto un canale, una linea di tendenza o un finanziamento orizzontale o livelli di resistenza. Le strategie che seguono il trend, come suggerisce il nome, includono le negoziazioni del giorno di apertura nella direzione della tendenza intraday sottostante. Il trend-following è probabilmente la strategia di trading più popolare tra i day trader perché restituisce un ottimo rapporto rischio/rendimento con un tasso di successo relativamente alto. Per aprire lo scambio sulla via del trend sottostante, attendi il valore per terminare un pullback (ad esempio, a un livello di Fibonacci intraday molto importante) e usa i modelli di candele per assicurarti che il trend sottostante sia sul punto di continuare.

Le strategie di trading in controtendenza includono l'apertura di operazioni nell'altro modo della tendenza sottostante. L'obiettivo del trader in controtendenza è catturare le correzioni di mercato che si verificano dopo un trend rialzista o ribassista cronico e potente.

Questa strategia di trading è leggermente più rischiosa del breakout trading e del trend following e può essere utilizzata solo da day trader esperti.

Altre opportunità di trading:

Poiché il day trading potrebbe essere uno stile di trading relativamente veloce, offre un numero enorme di opportunità di trading, su base giornaliera. I day trader basano le loro scelte totalmente su intervalli di tempo intraday, come quelli di 15 minuti, 30 minuti, 1 ora e 4 ore. Questi tempi offrono configurazioni molto più negoziabili rispetto ai grafici giornalieri o settimanali utilizzati dagli swing trader e dai position trader, il che potrebbe essere un grande vantaggio del day trading.

Tuttavia, tieni presente che i tempi a breve termine di solito contengono più rumore di mercato, che potrebbe accumulare rapidamente perdite se imposti i tuoi livelli di stop loss troppo stretti. Per evitare ciò, prova a misurare la volatilità tipica del titolo che stai negoziando (tramite l'indicatore ATR, ad esempio) e posiziona i tuoi stop loss per questo motivo.

Una quantità avanzata di possibilità di trading non significa essenzialmente più reddito. Un trader dovrebbe seguire il tuo piano di trading e piazzare solo quelle operazioni che sono completamente in linea con la tua strategia. Anche la gestione del rischio svolge un ruolo molto importante nel successo del day trading, quindi conferma di rischiare una piccola percentuale del tuo conto di trading su ogni singola operazione.

Costi di negoziazione più elevati:

Anche se fai day trading sul mercato, avrai costi di trading maggiori rispetto a una volta swing o position trading sul mercato. Poiché il day trading contiene l'apertura di più operazioni durante il giorno, scegli un broker con spread ridotti e commissioni di trading basse. Alcuni broker offrono spread stabili, che potrebbero essere entusiasmanti per i trader che vogliono negoziare attorno a importanti comunicati stampa e mantenere bassi i costi di trading. I comunicati stampa tendono a indirizzare verso un elevato slittamento del mercato, volatilità, costi di trading più elevati, che sono alcune cose che desideri comprendere se stai per negoziare importanti rapporti di mercato. A lungo termine, quei costi di trading possono sommarsi rapidamente e ridurre i tuoi profitti.

Potenziale di profitto limitato:

Supponendo i periodi di detenzione più brevi delle negoziazioni e i tempi più brevi su cui i day trader basano le loro scelte, il day trading include un potenziale di profitto più limitato legato allo swing trading. Inoltre, i trader chiudono le loro operazioni entro la fine del giorno di negoziazione, indipendentemente dal loro profitto. Sebbene questa pratica rimuova il rischio durante la notte, limita anche i probabili profitti di

promettenti assetti commerciali.

Rischio di indebitamento eccessivo delle tue operazioni:

La maggior parte dei mercati non cambia molto nel corso della giornata. Di conseguenza, i day trader utilizzano più leva per spremere i maggiori profitti e ottenere il massimo da queste piccole azioni sui prezzi. Sebbene la leva finanziaria sia spesso molto efficiente, i trader che sfruttano eccessivamente le loro operazioni rischiano anche perdite maggiori.

La leva finanziaria può essere un'arma a doppio taglio e verrà utilizzata solo per il tuo piano di trading per assicurarti di fare un rigoroso tentativo di gestione del rischio per limitare la tua influenza o rischio per operazione in modo tale da rimuovere il pericolo di rovina (es. account.)

Rumore di mercato

Più breve è il periodo di tempo, più fai trading sul rumore del mercato che devi gestire. Il rumore del mercato rappresenta un comportamento dei prezzi imprevedibile e imprevedibile senza alcun ragionamento tecnico o notizie che avrebbero portato a tali movimenti. Il rumore del mercato rappresenta un vero problema per i trader a breve termine, e quindi l'unico grazie per evitare di essere fermati troppo presto è ampliare il livello di stop-loss. Dai un'occhiata alla volatilità precedente all'interno della coppia e dai un'occhiata per allineare il tuo stop-loss sopra o sotto i recenti livelli di supporto e resistenza, dando al mercato spazio sufficiente per esibirsi.

CONCLUSIONE

Swing trading è estremamente popolare perché è amministrato su grafici con tempi più lunghi che consentono a un trader di scambiare i mercati, mantenere un impiego, studiare o fare altre cose con il proprio tempo. Può anche essere abituato a catturare le grandi mosse intraday per i trader che stanno cercando di fare trading all'interno delle sessioni e non vogliono effettuare operazioni durante la notte o per lunghi periodi di tempo. Questa è solo una strategia di trading dei tanti trader che possono avere la propria cassetta degli attrezzi. Prima di decidere se fa per te, conferma il tuo test e superalo su un conto di trading demo.

Un trader dovrebbe essere disciplinato e rigoroso per iniziare il day trading. Un tipico problema dei trader giornalieri è che agiscono e si discostano dalla loro strategia. A volte puoi passare prima che ti rendi conto che non stai seguendo rigorosamente la tua strategia iniziale. Ciò disturberà il tasso di vittoria della strategia e romperà le probabilità. Un onesto ringraziamento per affrontare i problemi di disciplina è quello di annotare le regole precise della tua strategia e attaccare la nota al tuo monitor in modo che sia sempre davanti a te durante le sessioni di trading. In questo modo, ti verrà continuamente ricordato di seguire le regole della tua strategia. In ciascuna delle operazioni di cui sopra, abbiamo calcolato con attenzione il risultato finale. Dovresti farlo anche tu, per avere dimestichezza con ciò che può accaderti esattamente in ogni operazione. Quando ti stabilizzi di più, diventa più facile, e anche alcune app di day trading progressive calcoleranno tutto spontaneamente per te. Sebbene sia diverso dal day trading, le recensioni e i risultati suggeriscono che lo swing trading potrebbe anche essere un sistema ingegnoso per i principianti con cui iniziare. Ciò può essere dovuto al fatto che l'intraday che cambia decine di titoli può rivelarsi troppo frenetico. Mentre gli swing trader vedranno i loro guadagni entro un paio di giorni, mantenendo alti i livelli di motivazione. Allo stesso tempo rispetto al trading a lungo termine, lo swing trading è abbastanza breve da fermare l'interruzione. mantenendo alti i livelli di motivazione. Allo stesso tempo rispetto al trading a lungo termine, lo swing trading è abbastanza breve da fermare l'interruzione. mantenendo alti i livelli di motivazione. Allo stesso tempo rispetto al trading a lungo termine, lo swing trading è abbastanza breve da fermare l'interruzione.

Il trading online è una delle parole più cercate su Google da chi ha intenzione di iniziare a guadagnare dai mercati finanziari. I dubbi sono vari. C'è la speranza di guadagnare soldi facili e di risolvere tutti i problemi economici in breve tempo e c'è anche il timore di perdere tutto a causa di qualche truffa.

Questi dubbi e malintesi nascono dal fatto che la maggior parte delle persone che sognano di fare trading non sanno veramente cosa sia.

Per molti è solo una vaga speranza di guadagno, qualcosa che genera soldi facili di cui hanno sentito parlare da qualche parente o amico. Il trading online, ovviamente, può essere un'attività estremamente redditizia a patto di avere una preparazione di base e, soprattutto, di utilizzare piattaforme serie,

Chi inizia a fare trading di opzioni binarie per la prima volta potrebbe trovarsi spiazzato dalla necessità di utilizzare strategie per guadagnare. L'idea di base è che una strategia di trading è qualcosa di estremamente complesso ed elaborato. Niente potrebbe essere più falso. Le strategie possono essere complesse ma possono anche essere semplici. Non usare mai strategie complesse se sei all'inizio della carriera di un trader. Ti consigliamo vivamente di fare solo le cose che puoi capire.

Una soluzione alternativa, perfetta per chi davvero non sa nulla di finanza, è costituita dai segnali di trading. Grazie a questi segnali è possibile copiare, in modo assolutamente automatico, le operazioni effettuate da trader esperti. In questo modo è possibile ottenere subito dei profitti anche se è evidente che tutto dipende dalla scelta iniziale che si fa